all rights reversed

HACKER

Απρίλιος 2015 # Τιμή τεύχους 3,99€

τεύχος 043 - Κόpavogur Edition



Δημιουργήστε τον δικό σας mailserver [*]

Έξυπνα κόλπα με το SSH

Εισαγωγή στα game mechanics

Προγραμματισμός με το .NET: File IO operations

Ισχυρή κρυπτογράφηση του email, απ' άκρο σε άκρο

[*] ...κι απαγκιστρωθείτε από τα μεγαθήρια

all rights reversed

HACKER


CMYK & WEB
cmykandweb.gr
GRAPHIC DESIGN
design • layout • web

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Κώστας Βεντούρας, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Πέτρος Κυλαδίτης, Αναστάσιος Μαγειρίας, Νίκος Μουρατίδης, Θωμάς Σερμπίνης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το ηλεκτρονικό περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στην Καλαμαριά Θεσσαλονίκης. Ο δικτυακός τόπος του περιοδικού είναι στο http://deltahacker.gr. Το email επικοινωνίας είναι το talk2us@deltahacker.gr και ειδικά για τις συνδρομές είναι το subscriptions@deltahacker.gr. Οι γενικές πληροφορίες για τις συνδρομές παρατίθενται στο http://deltahacker.gr/pdf12order και από την ίδια σελίδα γίνονται οι παραγγελίες συνδρομών.



Αυτό το Mac mini είναι σχεδιασμένο ώστε να κάνει ό,τι θέλετε – και πολλά περισσότερα.

Αυτό το Mac mini δεν είναι το βασικό μοντέλο, και **κοστίζει 1249 ευρώ** (http://store.istorm.gr/mac/mac-mini/2-8ghz-mac-mini.html).

Αυτό το Mac mini το **κληρώνουμε μεταξύ 25 αναγνωστών** με Jumbo-Pack Plus (60 τεύχη).

Αν η συνδρομή σας πλησιάζει στη λήξη της, αυτό το Mac mini **είναι εξαιρετικά πιθανό** να γίνει δικό σας.

Για να πάρετε μέρος στην κλήρωση, αρκεί να πάρετε κι εσείς **Jumbo-Pack Plus** συμπληρώνοντας κατάλληλα τη φόρμα παραγγελίας στο http://deltahacker.gr/pdf12order/#s2.

Από την ίδια τη φόρμα παραγγελίας βλέπετε **πόσους αναγνώστες με Jumbo-Pack Plus χρειαζόμαστε ακόμα**, ώστε να προχωρήσουμε στην κλήρωση.

Προσέξτε: Αν και **θα περιμένουμε όσο χρειαστεί** για να έχουμε 25 συνδρομητές με Jumbo-Pack Plus και να τους κληρώσουμε το Mac mini, αυτό δεν σημαίνει ότι δεν μπορούμε να κάνουμε κάτι για να επισπεύσουμε τη διαδικασία. Κάντε αυτό που πρέπει, συμπληρώνοντας τη φόρμα παραγγελίας: http://deltahacker.gr/pdf12order/#s2

Καλή σας επιτυχία!

Ο εκπαιδευτής εκπαιδεύεται

του @arkoudos

Αν και επισκέπτης, σας καλωσορίζω κι εγώ στο τεύχος 043 του deltaHacker! Όπως και τα προηγούμενα τεύχη, έτσι κι αυτό περιέχει γνώση και πληροφορία, από ανθρώπους που ενδιαφέρονται για το αντικείμενό τους κι έχουν αρκετό μεράκι ώστε να κάνουν το ενδιαφέρον τους επάγγελμα.

Κατ' εμέ, αυτό είναι θετικό. Κάθε μετάδοση πληροφορίας, είναι θετική. Θα κριθεί, θα αξιοποιηθεί, θα γίνει άλλο ένα σκαλοπάτι στην επίτευξη των στόχων ενός αναγνώστη.

Μολαταύτα, ως πρόσφατος πατέρας δύο μικρών –ακόμα– παιδιών, ομολογώ ότι ξαφνικά βρίσκω την τεχνολογία τρομακτική και τη μετάδοση της πληροφορίας επικίνδυνη. Ανατρέπω λοιπόν την ίδια μου τη φιλοσοφία και με αυτή την ανατροπή θα ήθελα να ξεκινήσω το εντιτόριαλ του τεύχους.

Ως πατέρας μιας τεχνολογικά προηγμένης γενιάς κι ως άνθρωπος που έζησε όλες τις φάσεις της ηλεκτρονικής ενηλικίωσης (το μόντεμ με χρονοχρέωση στην κλήση, το IRC κ.ά.), από τη στιγμή που έχω δει αρκετά σωστά αλλά και αρκετά στραβά της ολιγόχρονης –αλλά απόλυτα καθοριστικής– πορείας της τεχνολογικής προόδου, δεν νιώθω καθόλου, μα καθόλου προστατευμένος από το μέλλον που διαβλέπω.

Άνθρωποι που μοιράζονται προσωπικά δεδομένα αφειδώς στο Διαδίκτυο (όπως, π.χ., τον τηλεφωνικό τους αριθμό σε ένα comment – το έχω ζήσει πολλάκις), ουσιαστικά βάζουν μόνο το κερασάκι σε μία τούρτα προβλημάτων που οι ανεκπαίδευτοι (και δεν το λέω με κριτική διάθεση, θα εξηγήσω λίγο πιο κάτω τι εννοώ) θα ζήσουν μ' αυτά και τις συνέπειές τους. Στην ευκολία της μετάδοσης δεδομένων, που γίνεται ευκολία μετάδοσης *οποιουδήποτε* είδους δεδομένων, ακόμα και των πιο προσωπικών και πολύτιμων, η τεχνολογία δεν προσφέρει καμία απολύτως ασφάλεια, καθώς δεν μπορεί να ξεχωρίσει ποια από αυτά πρέπει / μπορούν να μοιράζονται, ποιωνο διαμοιρασμός θα ήταν καλό να αποφεύγεται, και ποια είναι σημαντικό να μην τα μοιράζονται ποτέ.

/var/log/messages

Η τεχνολογία δεν θα είναι ποτέ σε θέση να προσφέρει ένα δίχτυ ασφαλείας, όπως ένα όπλο δεν θα ξέρει ποτέ αν η σφαίρα βάλλεται εναντίον κάποιου που πρόκειται να σε πυροβολήσει ή εναντίον κάποιου αθώου. Η ευθύνη είναι πάντα στους χρήστες, στους χειριστές – και όχι στο μέσο. Αυτός που κρατάει το όπλο ή το εργαλείο, πρέπει να ξέρει και να το χρησιμοποιεί.



Θαυμάσια. Υπέροχα. Αν υπήρχε μόνο αυτό το πρόβλημα, θα γινόμουν εξαίρετος μπαμπάς. Όμως η τεχνολογία τρέχει πολύ γρήγορα και οι εξελίξεις είναι ραγδαίες. Δεν πήρα, π.χ., ποτέ χαμπάρι το SnapChat – ούτε καν το πότε βγήκε συνειδητοποίησα. Καταλαβαίνω ότι είναι πρακτικά αδύνατο να εκπαιδεύσω τα παιδιά μου *για κάθε* εργαλείο που θα κυκλοφορήσει, για κάθε εφεύρεση που θα χρησιμοποιήσουν, για κάθε εφαρμογή που θα πέσει στα χέρια τους. Για τα νέα δεδομένα, είμαι κι εγώ ανεκπαίδευτος. Η θέση μου μάλιστα ίσως είναι λίγο χειρότερη από εκείνη του διπλανού μου, ο οποίος άργησε ν' ανέβει στο τεχνολογικό τραίνο. Και το λέω αυτό διότι υπάρχουν στιγμές που νομίζω ότι ξέρω, χωρίς όμως στην πραγματικότητα να ξέρω.

"Κλείσε του τον υπολογιστή". "Μην του πάρεις κινητό". Εκτός του ότι έτσι θα κρατήσω το παιδί μου μακρυά από όλα τα θαυμάσια πράγματα που μπορεί να του προσφέρει η τεχνολογία, εκτός του ότι αργότερα θα κινδυνεύει να είναι εντελώς απροετοίμαστο για τον πραγματικό κόσμο, σε λίγο δεν θα έχω *καν* αυτή την ευκαιρία: Η "αθώα" τηλεόραση γίνεται έξυπνη και δικτυώνεται (και θα αποκτήσει, αν δεν έχει

ήδη, κάμερα για Skype), το αθώο παιχνίδι παραγίνεται έξυπνο (και αφουγκράζεται διαρκώς κι απαντά σαν τη Siri, βασιζόμενο σε τεχνολογίες όπως το Amazon Echo), το καημένο το ρολογάκι γίνεται προέκταση του τεχνολογικά υπερπροωθημένου κινητού κ.ο.κ. Αν προσπαθήσω να κρατήσω το παιδί μου τεχνολογικά αναλφάβητο για να το προστατέψω, ξέρω καλά ότι θα είναι αδύνατο να το καταφέρω. Η "δαιμόνια" τεχνολογία έχει εισχωρήσει παντού, ακόμα κι εκεί που δεν περίμενα: Η δασκάλα στο νηπιαγωγείο (δημόσιο, υπόψιν) χρησιμοποίησε το λάπτοπ της μαζί με τα παιδιά για να αναζητήσουν πληροφορίες, όλοι μαζί, για την ιστορία του παρακείμενου πάρκου. Μια θαυμάσια ενέργεια, που όμως θα εξάψει την περιέργεια της μικρής μου για τη χρήση του υπολογιστή. Το ίδιο θα συμβεί και αύριο που μία συμμαθήτριά της θα ανεβάσει, από το κινητό της, φωτογραφίες του απροστάτευτου παιδιού μου στο Facebook.

Η λύση, λοιπόν, του αποκλεισμού των τέκνων μου από τον κόσμο της τεχνολογίας, όσο κι αν φαντάζει "εύκολη" σίγουρα δεν είναι η ενδεδειγμένη. Και αυτό είναι μέρος του προβλήματος: Ότι το θέμα που αποφεύγω να παραδεχθώ είναι πως ναι μεν η τεχνολογία είναι το εργαλείο, ο άνθρωπος όμως χρειάζεται την εκπαίδευση κι εγώ, πέραν του ενστίκτου μου, δεν την έχω, για να την περάσω στα παιδιά μου.

Μεγαλώσαμε χωρίς κανόνες. Είναι βέβαια θαυμάσιο το ότι γαλουχηθήκαμε, σκοντάψαμε και πέσαμε, λερώσαμε τα γόνατά μας, αλλά ταυτόχρονα μάθαμε να φερόμαστε ως οι σκατζόχοιροι στο κρύο: ούτε πολύ κοντά για να τρυπηθούμε, ούτε πολύ μακριά για να κρυώνουμε. Μα αυτό που για μας ήταν μικρές, διόλου αυτονόητες κατακτήσεις, όπως π.χ. τα CMS για να αποκτήσουμε το δικό μας blog χωρίς να ξέρουμε να γράφουμε ούτε μια γραμμή κώδικα, για την επόμενη γενιά είναι υπερβολικά απλό κι ανώδυνα επικίνδυνο: Θα πατήσουν το Send, το Like και το Post *χωρίς* να έχουν την παραμικρή ιδέα τι πραγματικά σημαίνει αυτό. Και, ανάλογα με την ηλικία τους, θα είναι εξαιρετικά δύσκολο να τους εξηγήσεις τι σημαίνει αυτό: Πώς να πεις σε ένα μικρό παιδί ότι η αθώα κίνησή του μπορεί να σημαίνει πολλά για τον όχι-και-τόσο-αθώο αποδέκτη της;

Σας έδωσα την εντύπωση ότι δαιμονοποιώ την τεχνολογία; Ελπίζω πως όχι, δεν είναι αυτός ο σκοπός μου. Περισσότερο δείχνω τη δική μου ανικανότητα να μεταφέρω στα παιδιά μου τις βάσεις για την ορθή χρήση της. Όχι μόνο για να μη δεχθούν, π.χ., bullying από κανέναν, αλλά –ίσως και πολύ περισσότερο– για να μην το κάνουν εκείνα σε κάποιον άλλον. Όχι μόνο για να μην κάνουν αφελώς κακό στον εαυτό τους, αλλά για να μην κάνουν κακό –εξίσου αφελώς– και στους συνανθρώπους τους. Εγώ είμαι πρωτίστως ο ανεκπαίδευτος, εγώ είμαι αυτός που πρέπει να μάθει για να διδάξει μετά τους άλλους – έστω και τους μικρότερους.

Ήτοι, πριν διδάξω πρέπει να καθήσω στα θρανία. Και είμαι σχεδόν βέβαιος ότι το όλο θέμα δεν εξαντλείται στην τεχνολογική έκφανση του "καλώς φέρεσθαι" :)



Νιώθεις πραγματικά ασφαλής;
DANGER
UNAUTHORIZED PERSONNEL KEEP OUT
Ξεκίνα την προστασία σου
από το μηδέν
application security
infrastructure security
social engineering
incident response
projectzero.gr | 211 4053158
projectzero
IT security services and research

POST CARD

THE ADDRESS TO BE WRITTEN ON THIS SIDE



Ονέιλ μέιλ

Αγαπητοί μου,

Είχα υποσχεθεί στον εαυτό μου να σας στείλω την επιστολή Απριλίου μέσα στον Απρίλιο -το αργότερο Πρωτομαγιά-- αλλά ξέρετε τώρα πώς είναι αυτά τα πράγματα.

Εσύ δίνεις μιαν υπόσχεση, στον εαυτό σου ας πούμε, και το Σύμπαν --όπως λέει και ο ποιητής-- συνωμοτεί για να αποτρέψει την εκπλήρωσή της. Με τέτοια λιακάδα, Πρωτομαγιά μέρα, πώς να κάτσω μέσα να γράφω επιστολές, μου λέτε;

Τους είδα όλους να ετοιμάζονται για βόλτα και ζήλεψα.

Πήγαμε που λέτε Καισαριανή. Στο βουνό ψηλά, στη Μονή. Πρώτα περάσαμε από Σκοπευτήριο για να τιμήσουμε την ημέρα και μετά ανεβήκαμε με το λεωφορείο μέχρι όσο μας πήγαινε. Στο τέλος προχωρήσαμε με τα πόδια.

Αν εξαιρέσεις τα μυριάδες γυαλιστερά αυτοκίνητα των απεργών (η Πρωτομαγιά δεν είναι αργία, είναι απεργία -- remember;) η βόλτα μας ήταν πολύ όμορφη και αναζωογονητική.

Επιστρέψαμε προ ολίγου εξαντλημένοι και έχουμε όλοι πιάσει όλες τις ελεύθερες θέσεις στο σαλόνι και χαλαρώνουμε.

Εγώ, βεβαίως, το καθήκον καθήκον. Με το που μπήκαμε λοιπόν έβγαλα τη γραφομηχανή και ξεκίνησα να σας γράφω.

Είχα πάρει, που λέτε, προ ημερών το περασμένο τεύχος του περιοδικού μας στα χέρια μου και μ' ευχαρίστηση είχα δει πως το εντιτόριαλ του κου Αρχισυντάκτου αναφερόταν στην τελευταία μου επιστολή, αυτή στο τεύχος 042. Χάρηκα και ξεκίνησα να διαβάζω με ενδιαφέρον.

Αναρωτήθηκα --και αναρωτιέμαι-- πώς θα έπρεπε να αποκριθώ. Βλέπετε διστάζω. Για διάφορους λόγους.

Ο βασικότερος έχει να κάνει με τη διαφορετική μας αφετηρία.

Με το κείμενό του ο Α. απευθύνεται --το γράφει-- "στους εργαζόμενους, στους σπουδαστές, στους ανέργους", δηλαδή "σε όσους ζουν από τη δουλειά τους".

Έτσι, θεωρεί κάποια πράγματα δεδομένα. Θεωρεί, ας πούμε, πως οι άνθρωποι αυτοί έχουν νοιώσει "στο πετσί τους" τις επιπτώσεις της οικονομικής κρίσης (τη "φτωχοποίηση", όπως τη λέει), και πιστεύει πως γι' αυτόν ακριβώς το λόγο τα λεγόμενά του θα γίνουν πλήρως κατανοητά.

Αυτή είναι μια βασική μας διαφορά. Εγώ δεν είμαι ποτέ τόσο αισιόδοξος. Δεν είμαι ποτέ σίγουρος σε ποιόν απευθύνομαι και δεν θεωρώ καθόλου βέβαιο πως κάτι απ' όσα λέω μπορεί να γίνει πλήρως κατανοητό. Θα το έχετε προσέξει, αγαπητοί μου, πως η πιο συνηθισμένη μου αναψυχή, στις επιστολές αυτές, είναι να αποδομώ πράγματα που φαντάζουν αυτονόητα ή εύλογα.

Αναψυχή μεν, αλλά δεν το κάνω για χόμπι ή για πλάκα. Το κάνω διότι θεωρώ πως η τυφλή εμπιστοσύνη στους όρους είναι πολύ ύπουλη παγίδα.

Διαφωνώ, δηλαδή, με τον Α. (βλ. εντιτόριαλ 041), όταν μιλώντας για τις έννοιες (γενικά) έγραφε πως τις θεωρεί "καλά δομημένες κατασκευές της ανθρώπινης σκέψης". Not agree.

Να φέρω ένα παράδειγμα. "Εργαζόμενος". Τι σημαίνει ο όρος αυτός; Πώς τον ορίζουμε;

Ο εργάτης είναι εργαζόμενος. ΟΚ. Ο υπάλληλος του σούπερ μάρκετ επίσης. Ο προϊστάμενος βάρδιας είναι; Ο CEO της επιχείρησης; Ο εμπορικός αντιπρόσωπος; Ο πλασιέ; Ο χρηματιστής; Ο στρατιωτικός; Ο μουσικός; Ο δικαστής; Ο μεγαλογιατρός; Ο δημόσιος υπάλληλος;

Βλέπετε πού το πάω, έτσι; Όλοι αυτοί είναι εργαζόμενοι υπό την έννοια ότι όλοι ζουν από τη δουλειά τους. Δουλεύουν και πληρώνονται για τη δουλειά που κάνουν. Δεν πληρώνονται το ίδιο, ΟΚ, αλλά ούτε εισπράττουν προμήθειες ή τόκους ή ενοίκια. Δουλεύουν. Δεν κάθονται. (Αν προσέξατε δεν είπα ούτε "κτηματίας", ούτε "μεγαλοβιομήχανος", ούτε "εισοδηματίας", ούτε "γόνος καλής οικογένειας".)

Εργαζόμενοι λοιπόν όλοι τους, αλλά θα μου ήταν δύσκολο να πιστέψω πως όλοι αυτοί έχουν βιώσει το πετσί τους τη φτωχοποίηση -- ή έστω κάποια φτωχοποίηση με τον ίδιο τρόπο.

Τι κοινό, λοιπόν, έχουν όλοι αυτοί για τους οποίους γράφει ο Α., όταν λέει πως απευθύνεται σε όσους ζουν από την δουλειά τους; Δεν ξέρω.

Δεν ξέρω, κι αυτό με προβληματίζει. Αν οι όροι που χρησιμοποιούμε είναι τόσο ασαφείς και ευέλικτοι και σηκώνουν τόσο νερό, αντίστοιχα τόσο νερό θα σηκώνουν και τα συμπεράσματά μας.

Μπλέκουμε και δεν ξέρω πόσο εύκολα μπορούμε να ξεμπλέξουμε.

Για να το γενικεύσω λίγο.

Βγαίνει ο άλλος και σου μιλάει για την εργατιά. Η τιμημένη εργατιά, που δεν θα λυγίσει και λοιπά και λοιπά. Ρωτάω λοιπόν, εγώ. Ποια ακριβώς είναι η "εργατιά"; Τι σημαίνει αυτός ο όρος; Ποιους περιλαμβάνει; Πόσους αντιπροσωπεύει;

Αν το κοιτάξει κανείς λίγο, θα δει πως και αυτός ο όρος είναι ελαφρώς όλα και τίποτα.

Είναι ένας αδειανός όρος που γεμίζει με ό,τι τον γεμίσεις. Όλοι νομίζουμε πως ξέρουμε σε τι αναφέρεται, αλλά μόλις τον κοιτάξουμε από λίγο πιο κοντά γίνεται χαμός.

Η ας πάρουμε τον όρο "λαός". Τον σχολίαζα παλιότερα (επιστολή τεύχους 007). Βγαίνει ο καθένας και μας εξηγεί τι "θέλει" και τι "ζητάει" ο "λαός". Όλοι για το λαό μιλάνε, αλλά αν βάλεις δίπλα δίπλα αυτά που λένε, θα πάθεις κάτι.

Πασιφανείς αντιφάσεις μας αφήνουν όλους αδιάφορους, που λέει και ο ποιητής.

Γιατί; Γιατί ο καθένας από όσους μιλούν, ας πούμε για το λαό, προσποιούνται πως είναι σίγουροι για δύο πράγματα: ότι ξέρουν (οι ίδιοι) για ποιους μιλάνε, και ότι ξέρουμε (εμείς, οι ακροατές τους) για ποιους μιλάνε. Ενώ δεν ξέρουν.

Το πρόβλημα βέβαια είναι πως αυτή η ασάφεια είναι δομική.

Όλοι αυτοί οι όροι είναι όροι μπαλαντέρ. Όροι αδειανοί, σαν τα σακούλια, που ορίζονται μόνο σε αντιδιαστολή προς κάποιους άλλους, εξ ίσου "αυτονόητους", και γεμίζουν με ό,τι τους γεμίσουμε. Έχουν φτιαχτεί έτσι και αυτός ακριβώς είναι ο στόχος της ύπαρξής τους.



Παραδείγματα:

Ο λαός (vs οι ελίτ). Η κοινωνία (vs οι εχθροί της). Τα γηρατειά (vs τα νιάτα). Η νεολαία (vs το κατεστημένο). Η διεθνής κοινότητα (vs οι τοπικοί δικτάτορες). Τα αφεντικά (vs η εργατιά). Η εργατική τάξη (vs το κεφάλαιο). Ο Ελληνας (vs οι ξένοι). Οι κουτόφραγκοι (vs εμείς οι ξύπνιοι).

Και πάει λέγοντας.

Στη σύγχρονη πολιτική θεωρία τέτοιοι όροι ονομάζονται "κενά σημαίνοντα" και έχουν συζητηθεί αρκετά. Ψάχτε, για παράδειγμα, να δείτε τι λέει επ' αυτών ο Ερνέστο Λακλάου.

Το δικό μου πρόβλημα δεν περιορίζεται μόνο στον "αδειανό" τους χαρακτήρα (των κενών σημαινόντων).

Εχει να κάνει κυρίως με το άλλο μου αγαπημένο θέμα, αυτό της αδυναμίας διάκρισης επιπέδων - μετα-επιπέδων, που από ό,τι καταλαβαίνω παραμένει σκοτεινό παρόλες τις προσπάθειές μου.

Λοιπόν, παίρνω ως αφετηρία το εντιτόριαλ του τεύχους 042 και ξαναρχίζω.

Όταν μιλάμε για "κυρίαρχο τρόπο παραγωγής" αναφερόμαστε σε ένα περιγραφικό μοντέλο της εικαζόμενης εσωτερικής λογικής των μυριάδων οικονομικών συμβάντων που λαμβάνουν χώρα δίπλα μας. Δεν μιλάμε --το ξαναλέω-- για μια οντότητα του κόσμου τούτου.

Τίποτα από όσα μπορούμε να κάνουμε και να διεκδικήσουμε, είτε ως μεμονωμένα άτομα είτε ως οργανωμένες συλλογικότητες, τίποτα από όλα αυτά δεν μπορεί να αγγίξει τον "κυρίαρχο τρόπο παραγωγής". Ούτε να τον αγγίξει, ούτε να τον στηρίξει, ούτε να τον υποσκάψει. Τίποτα. Είναι σε άλλο virtual memory space, πώς να το πω.

Επιτρέψτε μου όμως να σας σκανδαλίσω λίγο, αγαπητοί μου.

Ο ίδιος ο όρος "εργατική τάξη" περιλαμβάνει τον "κυρίαρχο τρόπο παραγωγής" ως προϋπόθεση. Η "εργατική τάξη" υφίσταται μόνο κατά το μέτρο που υφίσταται ο "κυρίαρχος τρόπος παραγωγής". Εξ ορισμού. Κάθε, μα κάθε διεκδίκηση της εργατικής τάξης εμπεριέχει, εξ ορισμού, τη διατήρηση του κυρίαρχου τρόπου παραγωγής.

Ακόμα και σε μια σοσιαλιστική κοινωνία του ορατού μέλλοντος (λέμε τώρα) που τα μέσα παραγωγής θα είναι απολύτως κρατικοποιημένα και η εργατική τάξη θα προσφέρει αυτοβούλως και χαμογελαστά την υπεραξία της προς όφελος του κοινωνικού συνόλου (άλλο ένα κενό σημαίνον), ακόμα και τότε ο κυρίαρχος τρόπος παραγωγής θα είναι ο ίδιος:

Εργάτες θα μισθώνουν την εργασία τους, θα πληρώνονται για τον χρόνο τους, και θα αποξενώνονται από το προϊόν της εργασίας τους, το οποίο πλέον θα ανήκει στο κράτος.

Θα έρθει τώρα ο ινστρούχτορας και θα μου πει ότι αυτό θα είναι το ενδιάμεσο στάδιο, μέχρι να επέλθει η πλήρως αταξική κοινωνία.

Α, ναι, ΟΚ. Θαυμάσια. Ψώνιο. Αταξική κοινωνία. Τέλεια.

Να ένα ακόμα κενό σημαίνον. Κενό όσο και μεσσιανικό.

Αλλά θα επανέλθω.

Σας ασπάζομαι
Θείος Ακάκιος

Skill: Intermediate
Tags: mailserver, Mail User Agent, MUA, Mail Submission Agent, MSA, Mail Transfer Agent, MTA, Mail Delivery Agent, MDA, Mail Retrieval Agent, MRA, Domain Name System, DNS, blacklists, certificates

Ο δικός σας mailserver
Το γιατί και το πώς

Κάθε φορά που σχεδιάζουμε ένα νέο πρότζεκτ, η αλήθεια είναι ότι δεν χρειαζόμαστε πολλές δικαιολογίες για να βάλουμε μπρος: μας αρκεί να ξέρουμε ότι το πρότζεκτ θα αποδειχθεί ενδιαφέρουσα πρόκληση ή έστω ευχάριστη ενασχόληση. Εντάξει, σίγουρα μαθαίνουμε αρκετά στην πορεία, ενώ καθόλου δεν αποκλείεται να φτάσουμε και σε αποτελέσματα που είναι χρήσιμα στην πράξη. Αλλά ας μη γελιόμαστε: Αυτό που πραγματικά θέλουμε είναι να περνάμε καλά, αφού στην πλειονότητα των περιπτώσεων δεν υφίσταται καμία διάκριση μεταξύ πρότζεκτ και παιχνιδιού ;)

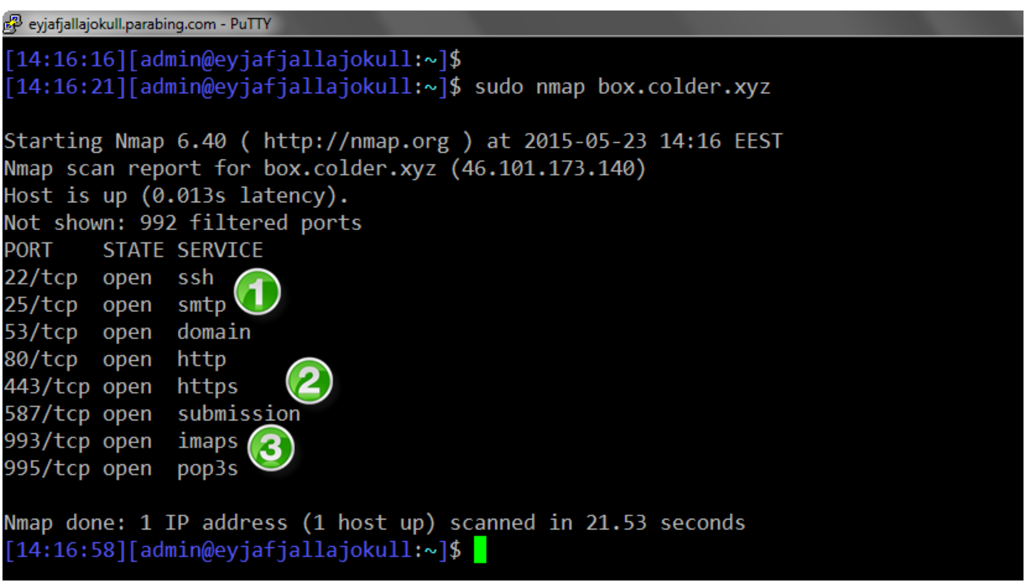
του Χρήστου Βαρελά

Σε μια πρώτη εξέταση, η δημιουργία ενός mailserver φαντάζει απλά ως άλλο ένα πρότζεκτ με το οποίο κάλλιστα θα μπορούσαμε ν' ασχοληθούμε – και είναι περίεργο που μέχρι τώρα δεν είχαμε ασχοληθεί. Θα το εντάσσαμε μάλιστα στην ίδια κατηγορία με πρότζεκτ όπως, π.χ., συγκρότηση του δικού μας router/firewall με το pfSense, στήσιμο του δικού μας OpenVPN server, δημιουργία webserver από το μηδέν κ.ά. Αναρωτιέστε τι κοινό έχουν τα παραδείγματα που μόλις αναφέραμε; Μα, το ότι όλα τους αφορούν σε προβλήματα που είναι ήδη λυμένα – και μάλιστα *καλά* λυμένα. Πράγματι:

- Αν για οποιονδήποτε λόγο επιθυμούμε ν' αναβαθμίσουμε την προστασία που παρέχει το modem/router μας στο τοπικό δίκτυο, τότε αντί να ψάχνουμε για νέο hardware που θα μπορέσει να δεχθεί το pfSense (http://deltahacker.gr/deltacast-s01e07) μπορούμε απλά να αντικαταστήσουμε τη συσκευή με μια καλύτερη. Εναλλακτικά, αν (σοφά) ζητάμε κάποια λύση Open Source, τότε το υπάρχον modem/router ίσως να δέχεται κάποιο εκ των OpenWrt/DD-WRT firmware.
- Αν θέλουμε παρουσία στο web δεν υπάρχει λόγος να ξεκινήσουμε με αγορά VPS, εγκατάσταση λειτουργικού συστήματος και ρύθμιση κάποιου webserver όπως, π.χ., είναι ο Apache ή ο nginx. Αντί για όλα αυτά, αρκεί να επιλέξουμε μια υπηρεσία web hosting και το νέο site απλά θα περιμένει από εμάς ν' αρχίσουμε ν' ανεβάζουμε περιεχόμενο.

- Αν πάλι χρειαζόμαστε πρόσβαση σε υπηρεσία VPN, τότε αντί να ψάξουμε για VPS και μετά να μπλέξουμε με το OpenVPN και τις ρυθμίσεις και τα κλειδιά και τους clients (http://deltahacker.gr/openvpn-revisited), αρκεί να στραφούμε σε κάποια από τις πάμπολλες υπηρεσίες VPN. Το μόνο που θα χρειαστεί να ελέγξουμε είναι αν η υπηρεσία κρυπτογραφεί ισχυρά και προστατεύει την ανωνυμία των πελατών της (βλ., π.χ., http://deltahacker.gr/which-vpn-service).
- Αν θέλουμε email και (ορθά) δεν μας κάνει αυτό που μάς έχει δώσει ο ISP, τότε αντί να στήσουμε τον δικό μας mailserver (είμαστε στο 2015, χελό-όου!) αρκεί ν' ανοίξουμε δωρεάν λογαριασμό σε κάποιο από τα Gmail, Yahoo!, Outlook κ.ά. Στοιχηματίζουμε μάλιστα ότι δεν υπάρχει αναγνώστης χωρίς *τουλάχιστον* μία διεύθυνση email σε Gmail, Yahoo!, Outlook, Live, Hotmail και τα ρέστα.

Προβλήματα σαν τα προαναφερθέντα είναι προ πολλού λυμένα, λοιπόν, αλλά ως αναγνώστες του deltaHacker ξέρετε πολύ καλά ότι, στην πράξη, συχνά γυρίζουμε την πλάτη στις έτοιμες λύσεις κι εφευρίσκουμε ξανά τον τροχό. Δεν είναι επειδή είμαστε (κρυφο)μαζοχιστές ή κάτι τέτοιο. Έχουμε όμως συνειδητοποιήσει ότι, αν θέλεις να μάθεις πραγματικά, τότε οφείλεις ν' ασχοληθείς, να πειραματιστείς, να φέρεις τα πάνω κάτω – ακόμη κι όταν εκ των προτέρων ξέρεις ότι η δική σου λύση *δεν* θα είναι καλύτερη από εκείνες που ήδη διατίθενται Εκεί Έξω (TM). Τώρα, εκτός από την εκπαιδευτική αξία της προσωπικής ενασχόλησης, σας ρωτάμε: Είναι το ίδιο διασκεδαστικό να φτιάχνεις κάτι μόνος σου, από το να το βρίσκεις έτοιμο; (Hint: Η σωστή απάντηση είναι "όχι, δεν είναι το ίδιο διασκεδαστικό" :P)



Κάπως έτσι φαίνεται μέσα από το nmap ο σημερινός μας mailserver (για το colder.xyz). Στο port 25/TCP συνδέονται άλλοι mailservers, μέσω SMTP (1), στο 587/TCP συνδέονται οι mail clients για να στέλνουν email, επίσης μέσω SMTP (2), ενώ στο 993/TCP συνδέονται οι mail clients για να λαμβάνουν email, μέσω IMAP (3).

Ζήτημα εμπιστοσύνης

Πιστεύουμε θα συμφωνήσετε πως το να υλοποιεί κανείς τις δικές του λύσεις είναι εκπαιδευτικό αλλά κι εξόχως διασκεδαστικό. Σας θέτουμε κι άλλο ένα ερώτημα: Εμπιστεύεστε εξίσου κάτι που σας προσφέρεται έτοιμο και λίγο-πολύ λειτουργεί ως μαύρο κουτί, όπως όσο εμπιστεύεστε κάτι παρόμοιο που έχετε φτιάξει μόνοι σας; Αναμφισβήτητα, αναλόγως για τι πράγμα μιλάμε η απάντηση διαφέρει...

Ειδικά στην περίπτωση του mailserver, το πρόβλημα της διακίνησης της ηλεκτρονικής αλληλογραφίας είναι εδώ και πολλές δεκαετίες λυμένο – και τη λύση αυτή υλοποιούν όμορφα και σωστά όλοι όσοι προσφέρουν υπηρεσίες email. Όμως το αρχικό ερώτημά μας δεν αφορά τόσο στην τεχνική αρτιότητα, όσο στον έλεγχο των δεδομένων που μας αφορούν και συχνά είναι εξαιρετικά ευαίσθητα. Αναδιατυπώνουμε, λοιπόν: Εμπιστεύεστε τα emails σας στους mailservers τρίτων; Ίσως ναι, ίσως όχι, ίσως δεν σας νοιάζει καν. Αν σας νοιάζει –έστω και λίγο– τι κάνετε γι' αυτό;

Μία λύση είναι η κρυπτογράφηση του email "από άκρο σε άκρο", π.χ. με χρήση του συνδυασμού Thunderbird & Enigmail. Όπως ενδεχομένως θα παρακολουθήσατε στα deltaCast s02e05 (http://deltahacker.gr/deltacast-s02e05) και s02e06 (http://deltahacker.gr/deltacast-s02e06), κάθε φορά που επιλέγουμε να κρυπτογραφούμε το email μας τότε ο μόνος που μπορεί να το διαβάσει είναι ο παραλήπτης και κανείς άλλος. Ας μένουν τα email μας αποθηκευμένα στους σκληρούς δίσκους των mailservers όσον καιρό (κάποιοι) θέλουν: Από τη στιγμή που είναι κρυπτογραφημένα, το μόνο που μπορεί να βλέπει κάποιος τρίτος είναι σωροί τυχαίων χαρακτήρων χωρίς κανένα νόημα απολύτως.

Κάπως έτσι είναι τα πράγματα με την κρυπτογράφηση δημοσίου κλειδιού, υπάρχει όμως κι ένα "αλλά": Ακόμη και στην ακραία περίπτωση που κρυπτογραφούμε *κάθε* email που στέλνουμε –μάλλον απίθανο, αλλά ας υποθέσουμε ότι συμβαίνει–, εξακολουθούμε να μην έχουμε τον έλεγχο του mailserver. Κι από τη στιγμή που δεν έχουμε τον έλεγχο του mailserver, 100% ήσυχοι δεν δικαιούμαστε να είμαστε. Για να μην κινδυνολογούμε, ας πούμε ότι είμαστε κατά 99,99% βέβαιοι ότι κανείς δεν πρόκειται να βγάλει νόημα από τα *κρυπτογραφημένα* μας emails. Όμως η NSA και άλλες υπηρεσίες ανά τον κόσμο αγωνίζονται γι' αυτό το 0,01%. Σύμφωνα με διαρροές που έχουν συμβεί κατά το παρελθόν, η NSA δείχνει ιδιαίτερο ενδιαφέρον *ειδικά* στις κρυπτογραφημένες επικοινωνίες. Ίσως σήμερα να μην είναι σε θέση να αποκρυπτογραφεί κρυπτογραφημένα μηνύματα όταν δεν έχει τα ιδιωτικά κλειδιά των παραληπτών, όμως συχνά φροντίζει να τα αποθηκεύει διότι...

- δεν αποκλείεται να αποκτήσει τα κλειδιά που χρειάζεται, π.χ., δια της πλαγίας ή/και της νομικής οδού
- ίσως μπορέσει να εκμεταλλευτεί κάποιο (άγνωστο) software bug, το οποίο επιτρέπει την κατάλυση του αντίστοιχου συστήματος κρυπτογράφησης.

Για να μη φτάνουμε σε υπερβολές, ας υποθέσουμε ότι τίποτε από τα προηγούμενα δεν συμβαίνει ή, ότι, ακόμη κι αν κάποια στιγμή συμβεί κάτι παρόμοιο, σιγά να μην είμαστε εμείς οι στόχοι :D Όμως το αρχικό ερώτημα επιμένει: Να εμπιστευόμαστε mailservers που δεν είναι υπό τον έλεγχό μας ή μήπως αξίζει να πάρουμε την κατάσταση στα χέρια μας;

Το email είναι το πιο σημαντικό στοιχείο της δραστηριότητάς μας στο Internet κι αν είστε από εκείνους που προτιμούν να έχουν τον έλεγχο, τότε συνεχίστε το διάβασμα. Σας ευχόμαστε από τώρα καλή διασκέδαση!

Τι ακριβώς είναι ένας mailserver;

Σε πρώτη προσέγγιση μπορούμε να υποθέσουμε –και δεν θα έχουμε άδικο– ότι πρόκειται για ένα εικονικό ή φυσικό μηχάνημα, το οποίο είναι εξοπλισμένο με το κατάλληλο λογισμικό ώστε να δέχεται και να διακινεί ηλεκτρονική αλληλογραφία. Κάπως έτσι έχουν τα πράγματα, μόνο που είναι αρκετά πιο περίπλοκα όσον αφορά στο λογισμικό. Επίσης, γι' αυτή τη διακίνηση της αλληλογραφίας χρησιμοποιούνται περισσότερα από ένα πρωτόκολλα. Αν λοιπόν θέλουμε μια καλύτερη εικόνα για το τι ακριβώς είναι ένας mailserver, τότε οφείλουμε να έχουμε κατά νου ότι στη συνολική υποδομή του email συναντάμε τις ακόλουθες κατηγορίες εφαρμογών.

MUA, Mail User Agent. Ονομάζεται και mail client και είναι ένα πρόγραμμα για τη λήψη και την αποστολή email. Παραδείγματα mail clients είναι τα Mozilla Thunderbird, Apple Mail, Microsoft Outlook κ.ά. Οι εφαρμογές web που παρέχουν πρόσβαση στο email μας, όπως αυτές των Gmail, Microsoft, Yahoo! κ.ά., αν και επίσης ανήκουν στην κατηγορία των MUA συνήθως αναφέρονται ως webmail applications.

MSA, Mail Submission Agent. Η δουλειά ενός MSA είναι να λαμβάνει τα emails που αποστέλλουν οι MUA και να τα δίνει σε κάποιον MTA για αποστολή (βλ. συνέχεια). Το επίσημο port για έναν MSA είναι το 587/TCP. Για την επικοινωνία με τους mail clients χρησιμοποιεί το πρωτόκολλο SMTP (Simple Mail Transfer Protocol) και απαιτεί authentication. Να σημειώσουμε εδώ ότι πολλοί MTA αναλαμβάνουν και υπηρεσίες MSA (από το port 25/TCP και χωρίς να απαιτούν authentication).

MTA, Mail Transfer Agent. Η ευθύνη ενός MTA είναι να διακινεί την ηλεκτρονική αλληλογραφία από υπολογιστή σε υπολογιστή, λειτουργώντας είτε ως client είτε ως server και πάντα ακολουθώντας το πρωτόκολλο SMTP. Το επίσημο port για την επικοινωνία μεταξύ των MTA είναι το 25/TCP. Το ίδιο port χρησιμοποιεί κι ένας MSA, όταν παραδίδει email σε κάποιον MTA (αλλά όταν παραλαμβάνει email από κάποιον MUA, ο MSA χρησιμοποιεί το port 587/TCP). Οι MTA ονομάζονται και mail relays και χωρίς αυτούς τα εξερχόμενα emails μας θα έφταναν το πολύ έως κάποιον MSA, αλλά όχι κατ' ανάγκη στους υπολογιστές των παραληπτών. Ένας υπολογιστής που τρέχει λογισμικό MTA αναφέρεται κι ως mailserver, mail exchanger ή MX host. Για την ηλεκτρονική αλληλογραφία, ο MTA αποτελεί ό,τι ένα ταχυδρομικό γραφείο αποτελεί για τη συνηθισμένη αλληλογραφία. Σε κάθε domain (π.χ., deltahacker.gr, parabing.com) είναι δυνατόν να αντιστοιχίζεται ένας mailserver, ο οποίος είναι υπεύθυνος για τη λήψη email που προορίζεται για το domain. Μπορούμε να φανταζόμαστε το domain ως μια περιοχή και τον MTA ως το ταχυδρομικό γραφείο που καλύπτει την περιοχή. Η αντιστοίχιση domain ↔ mailserver γίνεται στο επίπεδο του DNS με χρήση των λεγόμενων mail exchanger resource records ή απλά MX records. Μερικά παραδείγματα δημοφιλών MTA για Unixοειδή συστήματα είναι τα sendmail, qmail, Postfix, Exim και OpenSMTPD.

MDA, Mail Delivery Agent. Αποστολή ενός MDA ή διαφορετικά LDA (Local Delivery Agent) είναι να παραδίδει την αλληλογραφία στους *τοπικούς* χρήστες ενός υπολογιστή, αποθηκεύοντάς την στα αντίστοιχα mailboxes. Οι MDA λαμβάνουν την αλληλογραφία από τους MTA. Δημοφιλείς MDA για συστήματα Unix είναι τα procmail και maildrop.

MRA, Mail Retrieval Agent. Ο MRA είναι το λογισμικό που φέρνει την αλληλογραφία στους υπολογιστές των χρηστών. Συνεργάζεται με κάποιον MDA και αποτελεί είτε μεμονωμένη εφαρμογή (π.χ., fetchmail, getmail) είτε μέρος ενός MUA.

Σχηματικά, μια διευθέτηση όλων των προαναφερθέντων κατηγοριών λογισμικού θα μπορούσε να είναι η ακόλουθη:

MUA → MSA → MTA → ... → MTA → MDA ⇉ MRA ⇉ MUA

Τα απλά βελάκια (→) υποδηλώνουν λειτουργίες αποστολής email, ενώ τα διπλά (⇉) λειτουργίες λήψης. Το σχήμα αναπαριστά τη διαδρομή ενός μηνύματος από τον αποστολέα (MUA στ' αριστερά) έως τον παραλήπτη (MUA στα δεξιά). Σημειώνουμε ότι, κατά περίπτωση, ένας MTA ενδέχεται να εκτελεί και χρέη MSA. Παρατηρήστε επίσης το απλό βελάκι μεταξύ MTA και MDA: Ο τελευταίος στη σειρά MTA δεν είναι δυνατόν να αποθηκεύει τα emails που λαμβάνει. Δεν γνωρίζει πώς να το κάνει, ούτε είναι δική του ευθύνη, γι' αυτό και τα παραδίδει στον MDA. Εκείνος ξέρει πού και πώς να τα αποθηκεύει, οπότε δεν τα στέλνει κάπου αλλού. Με χαρά όμως δέχεται αιτήματα από τον όποιο MRA (διπλό βελάκι) και πρόθυμα τον εξυπηρετεί. Τέλος, ένας MUA είναι συνηθισμένο να ενσωματώνει και λειτουργίες MRA (αλλά όχι κατ' ανάγκη: βλ., π.χ., το Mutt).

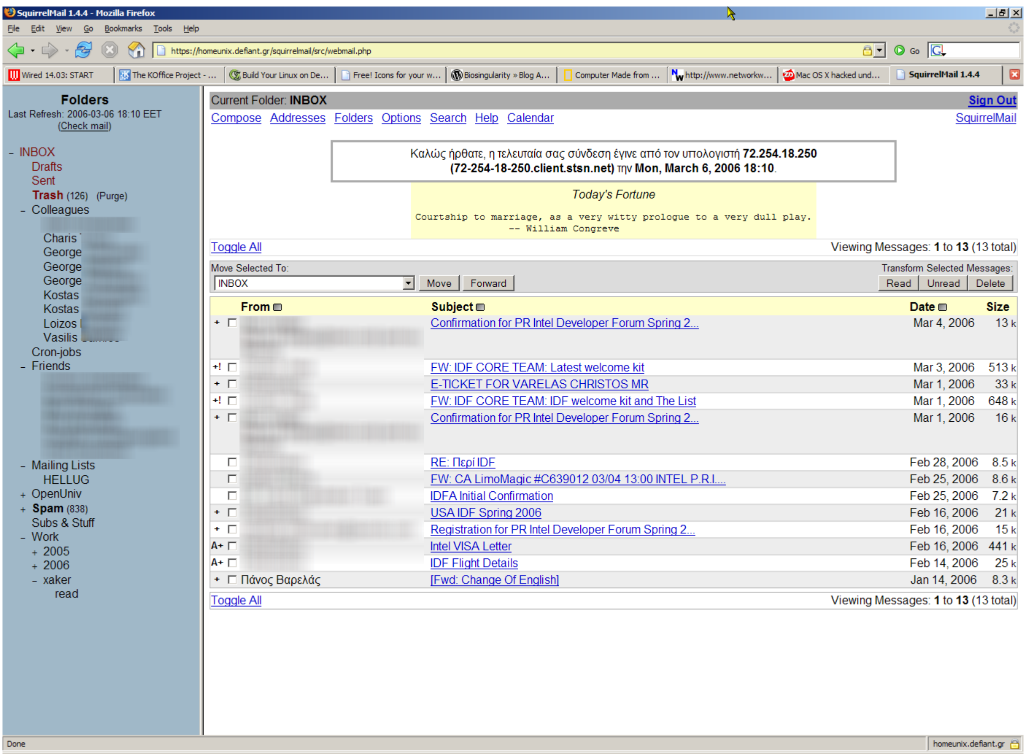
Μήπως τα μπλέξουμε;

Από την προηγούμενη ενότητα ίσως αποκομίσατε την εντύπωση ότι η δημιουργία ενός mailserver, ο οποίος θα λειτουργεί σωστά και αξιόπιστα, δεν είναι ό,τι πιο απλό. Αν πράγματι σας δημιουργήθηκε αυτή η εντύπωση, από τη μεριά μας ένα έχουμε να σας πούμε: Ορθά σας δημιουργήθηκε, οι mailservers είναι εξαιρετικά πολύπλοκες οντότητες :/

Και ξέρετε κάτι; Η κατάσταση είναι περισσότερο περίπλοκη απ' όσο αρχικά δείχνει. Ακόμη και μετά την επιλογή των κατάλληλων MxA και τη σωστή ρύθμισή τους, μένουν ένα σωρό άλλες δουλειές ώστε...

- τα credentials των λογαριασμών μας να μη μεταδίδονται σε μη-κρυπτογραφημένη μορφή
- τα emails μας να ταξιδεύουν μέσω κρυπτογραφημένων καναλιών
- το domain μας να μην καταλήξει σε blacklist με SPAM domains
- οι άλλοι mailservers να μη χαρακτηρίζουν τα emails που στέλνουμε ως SPAM (αυτό είναι δυνατόν να συμβαίνει κι ας μην είναι το domain μας σε blacklist).

Σκεφτείτε και το άλλο: Ως ο μοναδικός διαχειριστής του mailserver θα είστε κι ο μοναδικός υπεύθυνος για τη συντήρησή του ώστε, αν μη τι άλλο, να εφαρμόζονται έγκαιρα τα πλέον πρόσφατα security patches. Η αμέλεια στην περίπτωση ενός mailserver δεν συγχωρείται εύκολα. Αν λόγω κάποιου bug ή επιπόλαιης ρύθμισης ο mailserver πέσει στα χέρια των spammers ή αρχίσει να λειτουργεί ως open relay, τότε αφενός το μηχάνημά σας θα γίνει μέλος ενός δικτύου που συμβάλλει ώστε το Internet να μην είναι ένα τόσο ευχάριστο μέρος, αφετέρου το ωραιότατο domain σας θα καταλήξει σε ένα σωρό blacklists. Έστω κι αν μιλάμε για μία μόνο blacklist, το να αφαιρέσετε το domain σας απ' αυτή δεν θα είναι ό,τι πιο εύκολο.



*Εννέα χρόνια πριν: Έλεγχος του email μας από ξενοδοχείο στο San Fransisco, με σύνδεση στο web interface του mailserver μας, κάπου στο Νέο Κόσμο. Η επίσκεψη στο San Fransisco είχε γίνει στο πλαίσιο του IDF και τόσα χρόνια μετά δυσκολευόμαστε να θυμηθούμε πώς ακριβώς είχε καινοτομήσει *τότε* η εταιρεία :P*

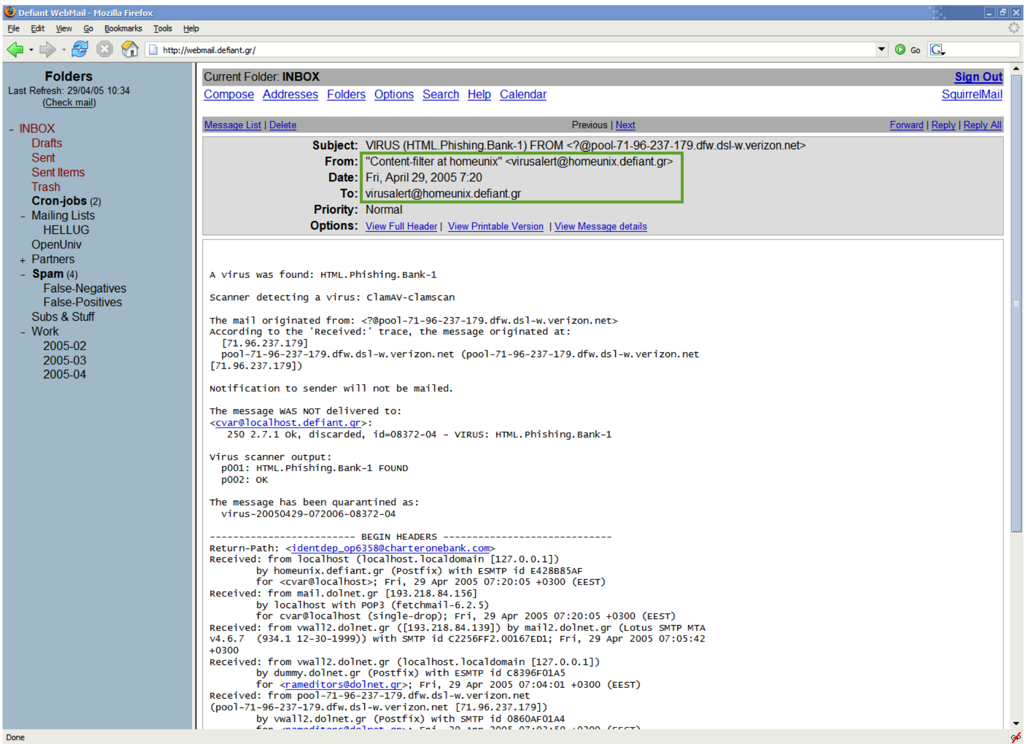
Μια φορά κι έναν καιρό

Παλιότερα, μιλάμε για τουλάχιστον 10 χρόνια πριν, συντηρούσαμε τον δικό μας mailserver σε ένα PC με Gentoo Linux, στο σπίτι. Από το 2003 διαθέταμε ένα domain, το defiant.gr, και το εν λόγω box είχε γίνει ο mailserver για το domain. Στο σπίτι βεβαίως δεν υπήρχε στατική διεύθυνση IP, οπότε χρησιμοποιούσαμε την υπηρεσία custom dynamic DNS της DynDNS (σήμερα ονομάζεται απλά Dyn) ώστε το hostname του box

να είναι το homeunix.defiant.gr. Το port 25, εξάλλου, ήταν μπλοκαρισμένο από τον ISP μας. Το ίδιο ισχύει και σήμερα με τους ISP, για λόγους περιορισμού του SPAM. Εξαιτίας αυτής της πολιτικής ο mailserver μας ήταν μη-προσβάσιμος από όλους τους άλλους mailserver. Γι' αυτό και είχαμε καταφύγει σε μια επί πληρωμή υπηρεσία που προσέφερε η DynDNS, η οποία αναλάμβανε να "μαζεύει" τα εισερχόμενα emails για το domain και να τα αποθηκεύει –για περιορισμένο χρονικό διάστημα– στους servers της εταιρείας. Το PC με το Gentoo δεν ήταν ανοικτό 24/7, αλλά όταν ενεργοποιούταν τότε η υπηρεσία της DynDNS του προωθούσε αυτομάτως τα email – και μάλιστα όχι στο port 25 αλλά σ' αυτό που είχαμε ορίσει εμείς (μάλλον ήταν το 50025). Ο MTA του box ήταν το postfix και, επιπρόσθετα, χρησιμοποιούσαμε το ClamAV για τον εντοπισμό ιών, καθώς και το SpamAssassin για το φιλτράρισμα της ανεπιθύμητης αλληλογραφίας. Την ίδια την αλληλογραφία κατά κύριο λόγο τη διαχειριζόμασταν μέσω του web interface που προσέφερε το SquirrelMail. Το όλο setup δούλευε μια χαρά και μάλιστα ο ταπεινός μας server είχε αποδείξει την αξία του με πολλούς τρόπους. (Πώς να το κάνουμε, το να είσαι τη μία στη Νέα Υόρκη, την άλλη στο Σαν Φρανσίσκο, την παράλλη στο Μόναχο και, σε κάθε περίπτωση, όταν επιστρέφεις στο ξενοδοχείο να παίρνεις το email σου από ένα PC κάπου στο Νέο Κόσμο, είναι ένα τεστ που μετράει.) Το μόνο θέμα που είχαμε με τον συγκεκριμένο mailserver αφορούσε στη συντήρησή του. Σκεφτείτε ότι το Gentoo ήταν –και συνεχίζει να είναι– source based distribution, οπότε από ένα σημείο και μετά οι μεταγλωττίσεις παύουν να είναι τόσο διασκεδαστικές και η μόνη τους θετική συνεισφορά είναι στην αύξηση του λογαριασμού της ΔΕΗ. Κάπως έτσι, το καλοκαίρι του 2007, όταν με τον κο Αρχισυντάκτη (TM) του deltaHacker βρισκόμασταν στο Τορόντο, αποφασίσαμε πως καλό θα ήταν, έτσι, για αλλαγή, να μεταφέρουμε το domain στη Google και το email μας να είναι, ουσιαστικά, hosted από τον "κολοσσό της αναζήτησης". (Έτσι λέγαμε τότε τη Google, κυρίως για να αυτοσαρκαστούμε με τις κλισέ εκφράσεις που χρησιμοποιούσαμε στο περιοδικό που γράφαμε.) Ένα μεσημέρι, λοιπόν, από το ξενοδοχείο που διαμέναμε, μετακομίσαμε το domain στο λεγόμενο "Google Apps for Domain" και με ένα σκριπτάκι μεταφέραμε όλη την αλληλογραφία από το PC στο Νέο Κόσμο, στο cloud της Google. Διακόψαμε βεβαίως την υπηρεσία mail forwarding (ή όπως αλλιώς λεγόταν) της DynDNS και, αφού βεβαιωθήκαμε ότι το email μας είναι πλέον προσβάσιμο μέσα από το γνώριμο web interface του Gmail, κάναμε κι ένα remote shutdown στο PC στο Νέο Κόσμο – ξέρετε, για οικονομία στο ρεύμα. (Ομολογουμένως, η αίσθηση που είχαμε περί οικονομίας την εποχή που η κρίση αποτελούσε, στην καλύτερη περίπτωση, θεωρητική κουβεντούλα μεταξύ τυρού και αχλαδίου, ήταν εντελώς διαφορετική από την αίσθηση που έχουμε σήμερα.)

Η μετάβαση στο Google Apps συνέβη ομαλά και το email του Defiant είναι έως σήμερα εκεί. Από το 2010 κι έπειτα φιλοξενούνται και τα emails άλλων domains μας στο Google Apps (deltahacker.gr & parabing.com) και, περιττό να σημειώσουμε, είμαστε απόλυτα ευχαριστημένοι. Μετά όμως από τις αποκαλύψεις Snowden, η αλήθεια είναι ότι το μικρόβιο της (σταδιακής) απεξάρτησης από τους μεγάλους cloud providers μάς έχει μολύνει για τα καλά. Εδώ και μήνες, το μόνο που μας κρατούσε από το να στήσουμε ξανά τον δικό μας mailserver, είχε να κάνει με την κρισιμότητα του email ως εφαρμογή: Δεν θέλουμε το παραμικρό downtime για τα emails μας, ειδικά για εκείνα που αφορούν στο deltahacker.gr. Συνυπολογίστε και το φαινόμενο "σκάει

η μία δουλειά μετά την άλλη, ο καιρός περνάει και δεν προλαβαίνω ν' ασχοληθώ με το πρότζεκτ που τόσον καιρό σκέφτομαι", οπότε καταλαβαίνετε γιατί έχουμε αναγγείλει το θέμα "ο δικός μου mailserver" από το προηγούμενο φθινόπωρο, αλλά μόλις πρόσφατα στρωθήκαμε κι ασχοληθήκαμε...



Ως λύση anti-virus για τον παλιό μας mailserver είχαμε το ClamAV, ο οποίος χρησιμοποιείται ακόμη και σήμερα.

Ομαλή κι ασφαλής προσέγγιση

Επειδή το downtime του email για τα domains μας δεν αποτελεί επιλογή και ταυτόχρονα δεν μπορούσαμε να καθυστερούμε άλλο το στήσιμο του δικού μας mailserver, αποφασίσαμε τελικά να διατηρήσουμε στο Google Apps τα defiant.gr, deltahacker.gr και parabing.com, αλλά να αποκτήσουμε ένα νέο (φθηνό) domain και να στήσουμε έναν mailserver γι' αυτό – βεβαίως από την αρχή. Η ιδέα είναι ότι θα χρησιμοποιούμε για αρκετό καιρό αυτόν τον mailserver, όχι όμως για emails που αφορούν στο περιοδικό. Εφόσον είμαστε αρκετά σίγουροι για τη σταθερότητα και την αξιοπιστία του, τότε εξετάζουμε και το σενάριο απαγκίστρωσης των άλλων domains από το Google Apps.

Μετά τόσα χρόνια από την τελευταία φορά που στήσαμε mailserver, δεν θέλαμε να επανέλθουμε στο θέμα ακολουθώντας την οδό του απόλυτου customization. Αποφασίσαμε έτσι να στραφούμε στο Mail-in-a-Box. Ουσιαστικά, πρόκειται για ένα

σύνολο από scripts που μετατρέπουν έναν cloud server (ή VPS) σε πλήρη mailserver. Σ' αυτόν τον mailserver έχουμε πρόσβαση από web (χάρη στο Roundcube) ή από οποιονδήποτε σύγχρονο MUA (μέσω IMAP). Το Mail-in-a-Box εγκαθιστά και ρυθμίζει έναν πλήρη DNS server, με όλες τις απαραίτητες εγγραφές για την υποστήριξη του email για το domain μας. Επιπρόσθετα, υλοποιεί μοντέρνα πρωτόκολλα (SPF, DKIM, DMARC) και πρακτικές ασφαλείας (opportunistic TLS, HSTS), ενώ προαιρετικά υποστηρίζει και το DNSSEC. Βεβαίως, παρέχει μηχανισμούς anti-spam (με το SpamAssassin) και greylisting (με το Postgrey), καθιστά εύκολη την εγκατάσταση πιστοποιητικών SSL (για τον webserver και τον mailserver), ενώ υποστηρίζει και το hosting στατικών web sites.

Είναι περιττό να τονίσουμε ότι η "χειροκίνητη" εγκατάσταση και σωστή ρύθμιση όλων των προαναφερθέντων υπηρεσιών και μηχανισμών ασφαλείας, σημαίνει τρομακτική δουλειά και μεγάλη πιθανότητα να φτάσουμε σε αποτέλεσμα που δεν θα δουλεύει όπως πρέπει. Ευτυχώς, το Mail-in-a-Box διευκολύνει απείρως τη δημιουργία ενός σύγχρονου, ασφαλούς και με πλούσια χαρακτηριστικά mailserver. Αρκετά όμως με τις συζητήσεις – ας πιάσουμε δουλειά!

Τι χρειαζόμαστε για τη συνέχεια

Η λίστα με τα απολύτως απαραίτητα για να ξεκινήσουμε, κάθε άλλο παρά μακροσκελής είναι.

- Χρειαζόμαστε ένα **VPS στο cloud**. Εξαιτίας του μεγάλου όγκου SPAM που διακινείται από οικιακούς υπολογιστές, είναι πρακτικά αδύνατον να στήσουμε mailserver στο σπίτι. Ακόμη κι αν ο ISP μας δεν μπλοκάρει κάποιο κρίσιμο port, οι οικιακές δημόσιες διευθύνσεις IP θεωρούνται ύποπτες για αποστολή SPAM και συνεπώς ο mailserver μας δεν θ' αργήσει να καταλήξει σε blacklist. Το VPS λοιπόν είναι μονόδρομος και θα πρέπει να έχει *τουλάχιστον* 768MB RAM. Εμείς φτιάξαμε ένα νέο droplet στη DigitalOcean, συγκεκριμένα το δεύτερο κατά σειρά μεγέθους, με 1GB RAM.

Σημείωση. Αν δημιουργήσετε το δικό σας droplet στη DigitalOcean ακολουθώντας αυτό το referral URL → http://bit.ly/digocean10off, τότε κερδίζετε αυτομάτως 10$ σε credit. Πρακτικά, για το droplet με 1GB RAM έχετε τον πρώτο μήνα δωρεάν κι από εκεί και πέρα –αν αποφασίσετε να συνεχίσετε– πληρώνετε 10$ το μήνα.

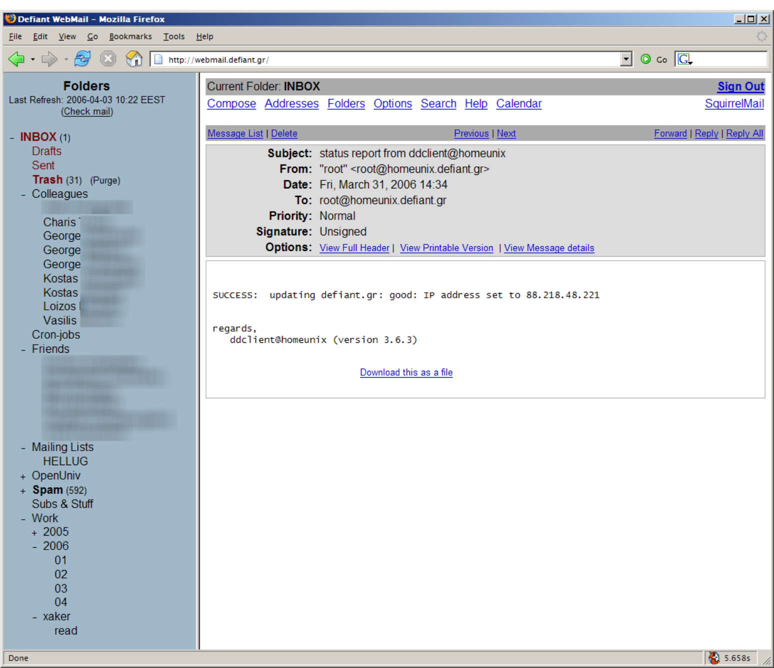
- Χρειαζόμαστε βεβαίως **ένα domain**, το email του οποίου θα εξυπηρετεί το Mail-in-a-Box. Ο registrar θα πρέπει να επιτρέπει τον ορισμό των λεγόμενων glue records ή αλλιώς child nameservers. Πρακτικά, τα glue records υποδηλώνουν ότι ο nameserver στο VPS μας γίνεται μέλος του παγκόσμιου domain name system. Ο registrar, εξάλλου, καλό θα ήταν να υποστηρίζει και το DNSSEC. Δεν είναι υποχρεωτικό, αν όμως παρέχεται και ρυθμιστεί τότε η ασφάλεια του box μας αναβαθμίζεται σημαντικά. Ένας registrar που **α)** υποστηρίζει τον ορισμό glue records, **β)** υποστηρίζει το DNSSEC, **γ)** προτείνεται από τον Joshua Tauberer, δημιουργό του Mail-in-a-Box (https://razor.occams.info), είναι το Gandi.net.

Σημείωση. Προτείνουμε με τη σειρά μας να επιλέξετε ένα οικονομικό domain από το Gandi.net, το οποίο θα υποστηρίζει το DNSSEC (δεν παρέχεται για όλα τα TLDs). Εμείς, π.χ., διαλέξαμε το **colder.xyz**, το οποίο κοστίζει 2,50 ευρώ ανά έτος. Για το κόστος κάθε TLD που παρέχει το Gandi δείτε το https://www.gandi.net/domain/price/info. Επίσης, στο http://wiki.gandi.net/en/domains/dnssec#who_can_use_dnssec δείτε όλα τα TLDs για τα οποία παρέχεται υποστήριξη DNSSEC από πλευράς Gandi.net.

- Αν σκοπεύουμε να χρησιμοποιήσουμε στα σοβαρά τον mailserver μας, τότε χρειαζόμαστε κι ένα κανονικό (όχι self-signed) πιστοποιητικό. Μπορούμε να αποκτήσουμε ένα δωρεάν από το http://www.startssl.com ή από το Gandi.net. Σε περίπτωση που πήραμε το domain μας από το Gandi.net, τότε δικαιούμαστε πιστοποιητικό SSL για το domain, δωρεάν για ένα χρόνο. Μετά τον πρώτο χρόνο, το πιστοποιητικό κοστίζει 16$ ανά έτος.

Μπορούμε να ξεκινήσουμε την εργασία μας χωρίς να έχουμε αποκτήσει κάποιο SSL certificate, όμως μετά την εγκατάσταση του Main-in-a-Box προτείνουμε την εγκατάσταση ενός κανονικού, όχι self-signed πιστοποιητικού. Αν δεν το κάνουμε, εκτός από τα warnings περί self-signed certificate (θα τα λαμβάνουμε τόσο από τον browser όσο και από τον όποιο MUA χρησιμοποιούμε), τα email μας ναι μεν θα φτάνουν στον προορισμό τους αλλά κατά πάσα πιθανότητα θα χαρακτηρίζονται ως SPAM.

Ας δούμε τώρα αναλυτικά πώς εγκαθιστούμε και ρυθμίζουμε το Mail-in-a-Box, ξεκινώντας από το VPS. Γυρίστε στη σελίδα 58.



Στο PC με το Gentoo Linux και τον mailserver μας τρέχαμε το ddclient, το οποίο επικοινωνούσε με την τότε DynDNS προκειμένου να ενημερώνει για αλλαγές στη δημόσια IP που παίρναμε από τον ISP μας. Σήμερα, εξαιτίας του γεγονότος ότι πολλοί οικιακοί υπολογιστές αποστέλλουν SPAM, η λειτουργία mailserver σε PC ή VM στο σπίτι μας είναι, πρακτικά, αδύνατη.

Skill: Beginner
Tags: game mechanics, gaming, game development

Game mechanics
Πρώτη προσέγγιση

Πίσω από κάθε video game που έχουμε παίξει και έχουμε κολλήσει, υπάρχει ένα μικρό σύστημα που το πολύ να έχουμε ακούσει ως αφηρημένη έννοια: τα game mechanics. Από το απλούστερο παιχνίδι στο κινητό μας μέχρι το νεώτερο τίτλο στην κονσόλα ενός gamer, τα game mechanics είναι ο συνολικός μηχανισμός ο οποίος, πίσω από τα γραφικά, το story και τους κανόνες, κάνει το παιχνίδι να "κινείται".

του Κώστα Βεντούρα

Τα game mechanics είναι, ως γενική έννοια, όλοι οι μηχανισμοί που επιτρέπουν στον παίκτη να "επικοινωνεί" με το παιχνίδι. Το κάθε όπλο που χρησιμοποιούμε, τα κρυμμένα αντικείμενα που ψάχνουμε, τα διαμάντια ή τα φρούτα που ταιριάζουμε σε σειρές για να τις εξαφανίσουμε, οι εχθροί που έρχονται σε "waves" κ.ο.κ., όλα αυτά είναι μέρος των game mechanics όπως τα έχουν σχεδιάσει οι developers.

Να ξεκαθαρίσουμε κάτι που έχει σημασία: Αν και τα game mechanics συνδέονται άμεσα με κάθε τμήμα του παιχνιδιού, μολαταύτα είναι ανεξάρτητα. Για παράδειγμα, ο στόχος ενός παιχνιδιού μπορεί να είναι να φτάσεις στο τέλος του δωματίου χωρίς να πέσεις σε παγίδες, όμως τα game mechanics αφορούν στο πώς λειτουργούν οι παγίδες, στο πώς τις αποφεύγουμε και στην κατανομή τους μέσα στο χώρο (τοποθετούνται τυχαία ή υπάρχει κάποια κρυμμένη λογική;). Επίσης, σε πολλά mobile games θα δείτε τον παίκτη να τρέχει ασταμάτητα και να προσπαθεί να αποφύγει εμπόδια που εμφανίζονται. Τέτοια παιχνίδια πιθανώς έχουν διάφορες εμφανίσεις (π.χ., ένας πιτσιρίκος που τον κυνηγάει ο φύλακας στο σιδηρόδρομο, ένας εξερευνητής που τρέχει μέσα σε δαιδαλώδεις ναούς κ.ο.κ.), όμως οι βασικοί τους μηχανισμοί τους είναι πανομοιότυποι.

Γιατί πρέπει να με ενδιαφέρουν;

Σαν παίχτης, δε θα χρειαστεί ποτέ να μπείτε στη διαδικασία να αναλύσετε τα mechanics ενός παιχνιδιού: Από τη φύση τους, είναι φτιαγμένα για να τα μαθαίνετε υποσυνείδητα καθώς παίζετε. Όταν είδατε για πρώτη φορά ένα πολεμικό FPS game, ίσως είχατε ξαφνιαστεί με το πόσο περίπλοκο φαινόταν: Αντίπαλα πυρά από όλες τις κατευθύνσεις, εκρήξεις παντού, η ομάδα σας να τρέχει και να μην τους προλαβαίνετε – και τόσα πολλά όπλα που δεν θυμάστε ποιο είναι ποιο και τι κάνει. Όμως μετά από κάποιο χρόνο γίνατε βετεράνος: ξέρατε πολύ καλά ποια όπλα σας

βολεύουν, πού να ψάξετε για να βρείτε πυρομαχικά, από πού να περιμένετε τους εχθρούς κ.ο.κ. Ξεφύγατε, δηλαδή, από το χάος των πληροφοριών κι αρχίσατε να διαβάζετε μόνο τους μηχανισμούς του παιχνιδιού που σας ενδιαφέρουν πραγματικά. Αυτό ήταν αναμενόμενο, αφού οι σχεδιαστές του παιχνιδιού έχουν πονοκεφαλιάσει, έχουν σχεδιάσει, έχουν αποτύχει κι έχουν ξανασχεδιάσει απ την αρχή, έως ότου το τελικό αποτέλεσμα να είναι ένα game που δε θα σας εκνευρίζει με το πόσο δύσκολο είναι στο χειρισμό ή στην κατανόηση της αποστολής.

Ας αρχίσουμε τώρα να εξετάζουμε το όλο θέμα από την οπτική ενός game developer: είτε απλά για να εκτιμήσετε την προσπάθεια των σχεδιαστών, είτε για να καταλάβετε πώς σκέφτονται και να γίνετε καλύτεροι gamers, είτε γιατί στην τελική μπορεί να φτιάξετε κι εσείς κάποιο παιχνίδι. Αν δεν το έχετε ήδη δοκιμάσει, μη σας φαίνεται και τόσο απίθανο. Αν μη τι άλλο, σήμερα υπάρχει μια πληθώρα από game engines, έτοιμα resources καθώς και sites/fora για να καθοδηγηθείτε, έτσι ώστε το να κατασκευάσετε ένα game τύπου Zelda δεν είναι ακατόρθωτο (αλλά ούτε κι εύκολο). Ειδικά στην τελευταία περίπτωση, ως φιλόδοξος game dev θα θέλετε να σχεδιάσετε ένα παιχνίδι που να είναι…

α) ελκυστικό προς τους παίκτες

β) κατανοητό στο χειρισμό του και

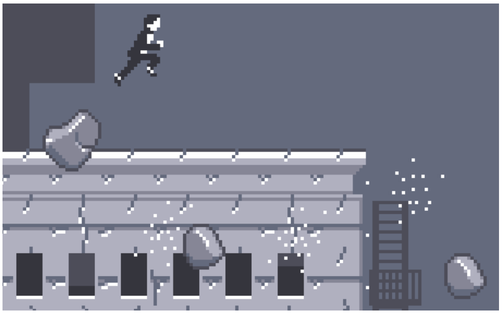
γ) πρόκληση για τον παίκτη (να τον κινητοποιεί προς κάποιο στόχο ώστε να συνεχίζει να παίζει – μ' άλλα λόγια να είναι εθιστικό).

Θα δείτε ότι το 90% της επιτυχίας σε αυτούς τους στόχους οφείλεται στη σωστή σχεδίαση των game mechanics.

Case studies

Ας δούμε κάποια παραδείγματα γνωστών και μη παιχνιδιών, καθώς και τους μηχανισμούς που κρύβονται από πίσω. Τα περισσότερα games βασίζονται σε μηχανισμούς που είναι γνωστοί σε όλους μας, υπάρχουν όμως και κάποια που ξεφεύγουν λιγότερο ή περισσότερο, υλοποιώντας πρωτότυπα κόλπα.

Canabalt (2009, http://www.adamatomic.com/canabalt).



Το Canabalt (2009) είναι ένα flash game που εντυπωσιάζει με την απλότητα των μηχανισμών και το "parkour" στυλ του.

Πρόκειται για ένα φαινομενικά απλό παιχνίδι, το οποίο όμως έχει κερδίσει αρκετούς θαυμαστές. Ο παίκτης διασχίζει τρέχοντας μια ατελείωτη πίστα από ταράτσες κτηρίων, κάνοντας ακροβατικά άλματα ώστε να αποφεύγει εμπόδια. Το παιχνίδι αυτό αναφέρεται ως πρώτο παράδειγμα διότι τα βασικά mechanics του είναι πολύ απλά. Η ιδέα των runner games ως γενικός μηχανισμός δεν είναι κάτι το καινούργιο, αφού ξεκίνησε από κλασικά racing games όπως το OutRun. Βασικό χαρακτηριστικό του Canabalt εί-

ναι η εισαγωγή της "άπειρης" πίστας, αφού το landscape δημιουργείται τυχαία καθώς τρέχουμε και άρα δεν τελειώνει ποτέ. Σκοπός είναι απλά να καλύψουμε όσο μεγαλύτερη απόσταση γίνεται, μέχρι να χτυπήσουμε εμπόδιο.

Kerbal Space Program (2011, https://kerbalspaceprogram.com).



Πώς να χτίσετε και να απογειώσετε ένα διαστημόπλοιο (μετά από πολλές προσπάθειες...)

Εδώ έχουμε ένα παράδειγμα περίπλοκων game mechanics. Στο απλούστερο mode του συγκεκριμένου παιχνιδιού (Sandbox mode, όπου δεν υπάρχουν περιορισμοί στους διαθέσιμους πόρους) ο χρήστης αναλαμβάνει να συνθέσει ένα διαστημικό σκάφος από διάφορα μέρη (προωθητές, δεξαμενές καυσίμου κο.κ.) και ύστερα να το απογειώσει και βεβαίως να το στείλει στο διάστημα. Το σύστημα εξομοίωσης είναι ιδιαίτερα φιλοσοφημένο υπολογίζοντας βαρυτικές δυνάμεις, ατμοσφαιρικές τριβές, συγκρούσεις, δομικές πιέσεις κ.λπ. με ικανοποιητικό ρεαλισμό. Το περιβάλλον με το οποίο ο παίκτης κατασκευάζει το σκάφος είναι αρκετά απλό, ενώ προβλέπεται ώστε ο χειρισμός του σκάφους εν πτήσει να προσφέρει συναρπαστική (καθώς και σχετικά ρεαλιστική – και σίγουρα δύσκολη) εμπειρία στον παίκτη. Τα mechanics του παιχνιδιού ακολουθούν τους γνωστούς κανόνες ενός space simulator, για όσους έχουν επαφή με το είδος (βλ., π.χ., το Orbiter). Θα προσέξετε ωστόσο ότι τέτοια παιχνίδια εστιάζουν είτε στην όμορφη απεικόνιση του διαστήματος, είτε στον μηχανισμό πτήσης. Το Kerbal Space Program προσπαθεί να πετύχει και στα δύο. Επίσης, η κατασκευή του σκάφους από επιμέρους τμήματα και η ελεύθερη τοποθέτησή τους σε όποιο σημείο του σκάφους θέλουμε, είναι μια σημαντική κι αξιοπρόσεκτη καινοτομία. Τέλος, για τους πιο φανατικούς, οι σχεδιαστές εισήγαγαν επιπλέον game modes στα οποία ο χρήστης έχει περιορισμένους πόρους για να κατασκευάσει το σκάφος του ή/και πρέπει να υλοποιήσει συγκεκριμένες αποστολές (προσγείωση σε άλλους πλανήτες, εγκαθίδρυση διαστημικών σταθμών και βάσεων κ.λπ.).

Minecraft (2011, https://minecraft.net).

Το πασίγνωστο Minecraft ανήκει στην κατηγορία των sandbox games. Ανήκει επίσης και στο κλαμπ των παιχνιδιών που γνώρισαν μεγάλη επιτυχία, ακριβώς λόγω των game mechanics. Στο παιχνίδι αυτό δεν θα συναντήσουμε εντυπωσιακά γραφικά, ούτε κάποιο συναρπαστικό backstory. Αντίθετα, ο χρήστης τοποθετείται μόνος μέσα σε έναν αχανή κόσμο από βουνά, θάλασσες και σπηλιές, όλα χτισμένα με κύβους (και με pixelated γραφικά). Ο παίκτης είναι σε θέση να εξερευνήσει τον κόσμο, να σκάψει, να συλλέξει κύβους από διάφορα υλικά και να κατασκευάσει άλλα αντικείμενα από αυτούς, καθώς και σπίτια ή άλλες κατασκευές. Η δημιουργία ενός παιχνιδιού όπου το περιβάλλον δύναται να καταστραφεί ή να αναπλαστεί από τον παίκτη, αποτελεί αρκετά παλιό concept (π.χ., τα Worms εισήγαγαν από πολύ παλιά το λεγόμενο "destroyable terrain"). Ωστόσο, σε συνδυασμό με τον κόσμο του Minecraft,

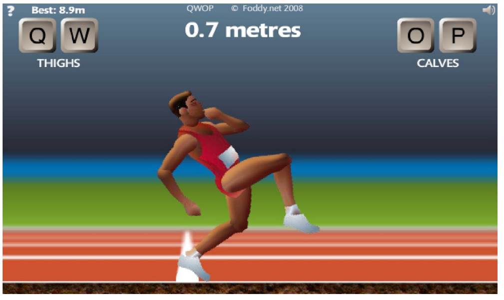
ο οποίος δημιουργείται τυχαία και είναι άπειρος σε έκταση (αναδημιουργείται καθώς ταξιδεύουμε σ' αυτόν), ο παίκτης έχει ένα τεράστιο playground στη διάθεσή του, το οποίο τον επιβραβεύει για την τάση του να εξερευνά. Επίσης, πέρα από απλές κατασκευές (π.χ., σπίτια) που μπορεί να φτιάξει με κύβους, ο παίκτης κατασκευάζει και αντικείμενα όπως σπαθιά και φτυάρια, κάνοντας συνδυασμούς υλικών και δοκιμάζοντας έτσι τη δημιουργικότητά του. Ένας ακόμα ενδιαφέρων μηχανισμός που προστίθεται στα παραπάνω είναι η εναλλαγή μέρας-νύχτας, όπου κατά τη διάρκεια της νύχτας ο παίκτης κινδυνεύει από τέρατα όπως αράχνες και σκελετούς κι έτσι ωθείται στο να χτίζει καταφύγια και να κατασκευάζει όπλα όπως σπαθιά και τόξα. Κάνοντας μια ανάλυση των επιμέρους game mechanics του παιχνιδιού, δεν είδαμε κάποια ιδιαίτερη καινοτομία σε κάποιο εξ αυτών. Ο ίδος ο συνδυασμός των διαφορετικών αυτών mechanics είναι αυτό που φέρνει την καινοτομία και προσελκύει τον παίκτη.



Στο Minecraft ταξιδεύουμε, σκάβουμε, χτίζουμε και εφευρίσκουμε μέσα σε έναν αχανή κόσμο φτιαγμένο από blocks.

QWOP (2008, http://www.foddy.net/Athletics.html).

Ένα πολύ ενδιαφέρον game από άποψη καινοτομίας, το QWOP σάς αναθέτει την τραγικά δύσκολη δουλειά του να κινήσετε ένα δρομέα στίβου ελέγχοντας τους μύες των ποδιών του. Χρησιμοποιώντας μόνο τα κουμπιά [Q], [W], [O] και [P], πρέπει να κάνετε τον αθλητή να τρέξει μια απόσταση 100 μέτρων. Περιττό να σας πούμε ότι απαιτούνται πολλές προσπάθειες και απάνθρωπο πείσμα, για να καταφέρει κάποιος έστω και μερικά βήματα. Η εξομοίωση διάφορων αθλημάτων συναντάται συχνά ως game mechanic (βλ., π.χ., τη μακροχρόνια σειρά παιχνιδιών Track & Field). Σε απλά παιχνίδια στίβου, ο σχεδιαστής συχνά θεωρεί ότι ο παίκτης μπορεί να τρέχει χωρίς καμία δυσκολία, πατώντας συνεχόμενα ένα πλήκτρο ή συνδυασμό πλήκτρων όσο πιο γρήγορα γίνεται, έχοντας αντιπάλους του παιχνιδιού για ανταγωνισμό (non-player characters, NPCs). Στο QWOP τα game mechanics διαφέρουν δραματικά: δεν υπάρχει λόγος για ανταγωνιστές, αφού η διαδικασία του βηματισμού είναι μια πρόκληση από μόνη της. Το παιχνίδι χρησιμοποιεί ragdoll physics (κάθε μέλος του σώματος κινείται σχεδόν ελεύθερα) και ο χρήστης ελέγχει μόνο τις γάμπες και τους αστραγάλους του δρομέα. Αν ο δρομέας χτυπήσει το έδαφος (με το κεφάλι), τότε game over. Προφανώς, από θέμα χειρισμού και ευκολίας το παιχνίδι αποτελεί παράδειγμα προς αποφυγή, ωστόσο ο σχεδιαστής αποσκοπεί στη διασκέδαση του παίκτη μέσω των αποτυχιών του.



Δύσκολα θα βρει κάποιος ένα παιχνίδι στίβου που να αποτελεί μεγαλύτερη πρόκληση απ' ό,τι το QWOP.

Brainstorming

Ας δοκιμάσουμε τώρα κάτι ενδιαφέρον. Είδαμε ότι πολλά games βασίζουν την επιτυχία τους στο σωστό συνδυασμό διάφορων μηχανισμών που χρησιμοποιούν. Σε τελική ανάλυση, οποιοσδήποτε συνδυασμός μηχανισμών μπορεί να οδηγήσει σε ένα ενδιαφέρον παιχνίδι – αν εκτελεστεί σωστά. Στον πίνακα που ακολουθεί βλέπουμε διάφορες κατηγορίες παιχνιδιών (κάθετα), οι οποίες είναι γνωστές στον μέσο gamer. Κάθε κατηγορία δοκιμάζουμε να τη συνδυάσουμε με κάποια game mechanics (οριζόντια), τα οποία έχουν δοκιμαστεί κι έχουν πετύχει σε πλήθος παιχνιδιών. Οι συνδυασμοί που προκύπτουν είναι άλλοτε απλοϊκοί κι άλλοτε αρκετά ενδιαφέροντες. Εσείς φυσικά μπορείτε να σκεφτείτε ακόμα καλύτερες ιδέες και να δοκιμάσετε περισσότερα mechanics για κάποιο project σας.

	RUNNING FOREVER	COLLECTING/ USING/ COMBINING ITEMS	STEALTH BASED	PHYSICS BASED	WITH VEHICLES
FIRST-PERSON GAME	Parkour game	Escape-the-room	*Thief* game	Break down walls etc.	Driving simulator
RACING GAME	*Tron* game	Build-your-own-car	Compete other thieves for a treasure	"Dirty" racing	Any racing game...
SHOOT-EM-UP GAME	*Time Crisis* clone?	Fight with customizable guns	Sniper game	Catapult game	Tanks game
STRATEGY GAME	Control routes of non-stop railways	Research and build robot armies	Control anti-terrorist squad	Control bulldozers to clean up dumps	Parking cars game
ADVENTURE GAME	Find way out of collapsing building	*Sierra*-style adventure	*Metal Gear Solid* clone	*Amnesia*-style adventure	Solve a mystery in a moving train
SURVIVAL GAME	Escape zombie horde	Survive on deserted island	Survive on deserted island with cannibals	Tower defense game	Fix car to escape from zombie infestation

Συμπερασματικά

Ακολουθούν ορισμένα πράγματα που είναι χρήσιμο να έχει κάποιος κατά νου, αν δουλεύει πάνω σε game project.

- Όσο ασυνήθιστο κι αν είναι κάποιο game, οφείλει να χρησιμοποιεί μηχανισμούς που ο παίκτης γνωρίζει από προηγούμενη εμπειρία ή μπορεί εύκολα να μάθει.

- Όσο πιο περίπλοκους μηχανισμούς χρησιμοποιούμε, τόσο οφείλουμε να ελέγξουμε ότι λειτουργούν σωστά και παρέχουν ομαλό gameplay σε όλες τις περιπτώσεις.
- Δώστε σε τρίτους, π.χ., σε φίλους σας ή σε γνωστούς από fora, την ευκαιρία να δοκιμάσουν το παιχνίδι σας πριν το διαθέσετε στο ευρύ κοινό, ώστε να ξετρυπώσετε προβλήματα ή δυσκολίες στους μηχανισμούς που ενδεχομένως δεν παρατηρήσατε.
- Γενικά, προτιμήστε να μη βασίζετε το σχεδιασμό όλου του game πάνω σε κάποιο mechanic που οπωσδήποτε θέλετε να χρησιμοποιήσετε. Για παράδειγμα, ίσως θέλετε να φτιάξετε κάποιο game βασισμένο σε physics, αλλά αν δεν έχετε κάποιο καλό concept για το παιχνίδι ή αν δεν μπορείτε να το συνδυάσετε με physics, μη βιαστείτε. Σκεφτείτε λίγο περισσότερο πώς θα το βελτιώσετε ή δοκιμάστε κάτι άλλο.
- Σκεφτείτε το κοινό στο οποίο απευθύνεστε καθώς σχεδιάζετε τα mechanics σας. Αν, π.χ., φτιάχνετε ένα mobile game, πρέπει να σκεφτείτε τους περιορισμούς των κινητών συσκευών (π.χ., μικρές οθόνες) αλλά και τις τάσεις των mobile gamers (π.χ., προτίμηση προς casual παιχνίδια), έτσι ώστε να προσαρμόσετε το παιχνίδι σας κατάλληλα.
- Επιλέξτε μηχανισμούς που μπορείτε να υλοποιήσετε. Αν, π.χ., θέλετε να χρησιμοποιήσετε physics στο παιχνίδι αλλά δεν έχετε δουλέψει ποτέ με physics, κάντε πρώτα κάποια δοκιμαστικά πειράματα για να εξοικειωθείτε *πριν* ξεκινήσετε μεγαλύτερα projects.

Όπως πάντα, η εμπειρία θα έχει να σας διδάξει πράγματα που κανένα άρθρο δεν μπορεί να σας διδάξει. Γι' αυτό, αν έχετε ήδη σκεφτεί να δοκιμάσετε τις ικανότητές σας ως game designers, μη διστάσετε. Μετά από μερικές προσπάθειες και με κατάλληλη έμπνευση, θα δείτε ότι θα κάνετε σημαντική πρόοδο. Και σε όσα εμπόδια συναντήσετε, το documentation και τα help fora είναι οι καλύτεροί σας φίλοι. Καλή τύχη!

Ο Κώστας Βεντούρας ασχολείται με game development και web design. Είναι συντάκτης στο JunkBytes (http://www.junkbytes.com), μαζί με τον Αναστάσιο Μαγειρία.

Skill: Intermediate
Tags: .NET, C#, programming, console, io

Ανάπτυξη εφαρμογών σε C# για το .NET
[μέρος 5ο]

Έφτασε η ώρα να αφήσουμε πίσω την είσοδο δεδομένων από το πληκτρολόγιο και την εμφάνιση της εξόδου στην εκάστοτε κονσόλα. Σ' αυτό το μέρος της σειράς μας θα προχωρήσουμε στην αλληλεπίδραση με το σύστημα αρχείων και θα εξερευνήσουμε τις δυνατότητες που προσφέρει το .NET framework, για την είσοδο και την έξοδο δεδομένων σε αρχεία.

του Πέτρου Κυλαδίτη

Όλα τα προγράμματα, από τα πιο μικρά κι απλά μέχρι και τα πιο μεγάλα και σύνθετα, αποτελούν "μηχανές" που επεξεργάζονται δεδομένα. Η πληροφορία εισέρχεται σε αυτές τις μηχανές από κάποια "συσκευή εισόδου" και, όταν ολοκληρωθεί η επεξεργασία, κατευθύνεται σε κάποια "συσκευή εξόδου". Ως γνωστόν, η τυπική συσκευή εισόδου (standard input device) είναι το πληκτρολόγιο, ενώ η τυπική συσκευή εξόδου είναι η οθόνη. Αυτό βέβαια ισχύει για τα πολύ απλά προγράμματα και συνήθως γι' αυτά με τα οποία ξεκινά κανείς τη γνωριμία του με μια γλώσσα προγραμματισμού. Ε, λοιπόν, τα προγράμματα που έχουμε δει μέχρι στιγμής ανήκουν σε αυτήν ακριβώς την κατηγορία. Σε αυτό το άρθρο θα ανέβουμε επίπεδο (level up!) και θα δούμε μερικά προγράμματα που χρησιμοποιούν ως είσοδο αλλά κι ως έξοδο κάποια αρχεία. Ουσιαστικά, θα γνωρίσουμε τις κλάσεις και τις μεθόδους που παρέχει το .NET Framework για το χειρισμό των αρχείων – και του συστήματος αρχείων γενικότερα. Κι επειδή θα ήταν περιττό να σχολιάσουμε τη σπουδαιότητα αυτής της γνωριμίας, θα μπούμε κατευθείαν στο ψητό. Οι κλάσεις που θα μας απασχολήσουν ανήκουν στο namespace με όνομα System.IO.

Εγγραφή δεδομένων

Η εγγραφή δεδομένων σε ένα αρχείο επιτυγχάνεται με τη βοήθεια των στατικών μεθόδων της κλάσης File. Ας ρίξουμε μια ματιά στη συμπεριφορά καθεμιάς.

- **File.WriteAllText(string path, string data).** Με αυτή τη μέθοδο τα δεδομένα που περιέχει η συμβολοσειρά data εγγράφονται στο αρχείο που δηλώνει η παράμετρος path. Αν η συγκεκριμένη μεταβλητή δεν περιλαμβάνει κάποια απόλυτη διαδρομή (absolute path), τότε αντιμετωπίζεται ως σχετική ως προς τη θέση του εκτελούμενου αρχείου.

- **File.WriteAllLines(string path, string[ ] data).** Αυτή τη φορά τα προς εγγραφή δεδομένα βρίσκονται σε έναν πίνακα. Η τιμή από κάθε κελί του πίνακα αποθηκεύεται σε μια ξεχωριστή γραμμή του αρχείου.
- **File.WriteAllBytes(string path, byte[ ] data).** Αυτή η μέθοδος αποθηκεύει και πάλι τα δεδομένα ενός πίνακα. Αυτή τη φορά, όμως, δεν χρησιμοποιούνται διαφορετικές γραμμές και τα δεδομένα γράφονται σε διαδοχικές θέσεις, byte προς byte. Όπως μπορούμε εύκολα να μαντέψουμε, η WriteAllBytes χρησιμοποιείται για την εγγραφή αρχείων με δυαδικά δεδομένα (binary data).
- **File.AppendAllText(string path, string data).** Η AppendAllText λειτουργεί όπως και η WriteAllText, με τη διαφορά ότι δεν επικαλύπτει τα δεδομένα που ενδέχεται να περιλαμβάνει ήδη το δοθέν αρχείο. Τα νέα δεδομένα προστίθενται στο τέλος του αρχείου.

Η εγγραφή δεδομένων στο δίσκο εγκυμονεί διαφόρους κινδύνους. Υπάρχουν αρκετοί λόγοι για να πάει κάτι στραβά. Για παράδειγμα, μπορεί ο χρήστης να μην έχει δικαιώματα εγγραφής σε κάποια θέση του συστήματος αρχείων, το όνομα του αρχείου να μην είναι αποδεκτό, ο χώρος του δίσκου να έχει εξαντληθεί, να εκδηλωθεί κάποια αστοχία στο μέσο αποθήκευσης κ.ο.κ. Αν συμβεί κάτι απ' όλα αυτά, θα προκληθεί μια εξαίρεση. Το πρόγραμμά μας πρέπει να είναι κατάλληλα προετοιμασμένο γι' αυτή την περίπτωση, ώστε να μη διακοπεί η ομαλή λειτουργία του.



Η εγγραφή και η ανάγνωση αρχείων δεν ήταν πάντα ό,τι πιο εύκολο. Όσοι πρόλαβαν αυτές τις εποχές, πρέπει να αισθάνονται πανευτυχείς με τις δυνατότητες που προσφέρουν οι σύγχρονες γλώσσες προγραμματισμού.

Επειδή έχουμε βάσιμες υποψίες ότι μας κατέκλυσε ακατάσχετη λογοδιάρροια και καθώς δεν είναι του χαρακτήρα μας να αρκούμαστε μόνο στη θεωρία, θα εξετάσουμε τώρα μερικά παραδείγματα. Θα ξεκινήσουμε με την WriteAllText. Το πρόγραμμά μας θα αποθηκεύει το όνομα του χρήστη στο αρχείο name.txt, στην ίδια θέση με το εκτελέσιμο. Ανοίγουμε λοιπόν το αγαπημένο μας SharpDevelop, δημιουργούμε ένα νέο project εφαρμογής κονσόλας και εισάγουμε τον ακόλουθο κώδικα:

```
01 using System;
02 using System.IO;
03
04 namespace FileSystem {
05   class Program {
06     public static void Main( string[] args ) {
07       Console.WriteLine( "Πως σε λένε;" );
08       string usrin = Console.ReadLine();
09
10       try {
11         File.WriteAllText( "name.txt", usrin );
12         Console.WriteLine( "Το όνομά σου αποθηκεύτηκε επιτυχώς στο αρχείο
           'name.txt'" );
13       } catch( Exception error ) {
14         Console.WriteLine( error.Message );
15       }
16       Console.ReadKey( true );
17     }
18   }
19 }
```

Στη γραμμή 02 δηλώνουμε τη χρήση του namespace System.IO. Όπως είπαμε ήδη, το συγκεκριμένο namespace περιλαμβάνει τις κλάσεις για το χειρισμό των αρχείων. Νομίζουμε ότι μέχρι και τη γραμμή 08 τα πάντα είναι γνώριμα και απλά. Το πρόγραμμα τυπώνει ένα μήνυμα προτροπής και τα δεδομένα που θα εισαγάγει ο χρήστης θα αποθηκευτούν στη μεταβλητή usrin. Στη γραμμή 10 ξεκινάει ένα block τύπου try, που το χρησιμοποιούμε για την αντιμετώπιση των πιθανών εξαιρέσεων. Μέσα στο block συναντάμε τη χρήση της μεθόδου WriteAllText, με την οποία αποθηκεύουμε την τιμή της μεταβλητής usrin, στο αρχείο name.txt. Στη συνέχεια (γραμμή 12) εμφανίζουμε ένα μήνυμα επιτυχίας. Εδώ χρειάζεται λίγη προσοχή: Το συγκεκριμένο μήνυμα δεν θα εμφανιστεί καθόλου, αν κατά την προσπάθεια εγγραφής του αρχείου προκύψει κάποιο πρόβλημα. Σε αυτή την περίπτωση η ροή του προγράμματος θα μεταφερθεί αμέσως στο μπλοκ catch. Μάλιστα, το σύστημα θα τροφοδοτήσει το συγκεκριμένο μπλοκ με μια παράμετρο που, όπως μπορείτε να δείτε στη γραμμή 13, αποτελεί ένα αντικείμενο τύπου Exception. Στη γραμμή 14 φροντίζουμε για την ενημέρωση του

χρήστη, με την εμφάνιση του σφάλματος που προέκυψε. Το σχετικό μήνυμα το λαμβάνουμε από την ιδιότητα Message, του αντικειμένου Exception που παρείχε το σύστημα. Κατά τα γνωστά, στη γραμμή 16 χρησιμοποιούμε τη μέθοδο Console.ReadKey, για να εισάγουμε μία παύση στην εκτέλεση του προγράμματος.

Με τη χρήση της WriteAllText, κάθε φορά που θα εκτελούμε το πρόγραμμα το περιεχόμενο του αρχείου name.txt θα αντικαθίσταται από μια νέα συμβολοσειρά. Στην περίπτωση που θα θέλαμε να διατηρούνται τα υπάρχοντα δεδομένα, θα ήταν αρκετό να αντικαταστήσουμε τη WriteAllText με την AppendAllText. Ας προχωρήσουμε τώρα σε ένα ακόμα παράδειγμα, με τη WriteAllLines. Αυτό το πρόγραμμα θα κατασκευάζει ένα τμήμα του πίνακα της προπαίδειας, ανάλογα με τον αριθμό που εισάγει ο χρήστης. Το περιεχόμενο του πίνακα θα αποθηκεύεται στο αρχείο prop-X.txt, με το X να αποτελεί τον αριθμό που έδωσε ο χρήστης.

```
using System;
using System.IO;

namespace Propaideia {
  class Program {
    public static void Main( string[] args ) {
      Console.Write( "Πίνακας προπαίδειας για τον αριθμό: " );
      string usrin = Console.ReadLine();
      int usrnum;
      try {
        usrnum = int.Parse( usrin );
      } catch( Exception err ) {
        Console.WriteLine( err.Message );
        Console.ReadKey( true );
        return;
      }

      string[] prop = new string[11];
      for( int i=0; i<prop.Length; i++ ) {
        int apotelesma = usrnum * i;
        prop[i] = i.ToString() + " x " + usrnum.ToString() + " = " + apotelesma.ToString();
      }

      string filename = "prop-" + usrnum.ToString() + ".txt";
      try {
        File.WriteAllLines( filename, prop );
```

```
27         Console.WriteLine( "Η προπαίδεια για το " + usrnum + " αποηθεύτηκε
           επιτυχώς στο αρχείο '" + filename + "'" );
28       } catch( Exception err ) {
29        Console.WriteLine( err.Message );
30       }
31       Console.ReadKey( true );
32     }
33   }
34 }
```

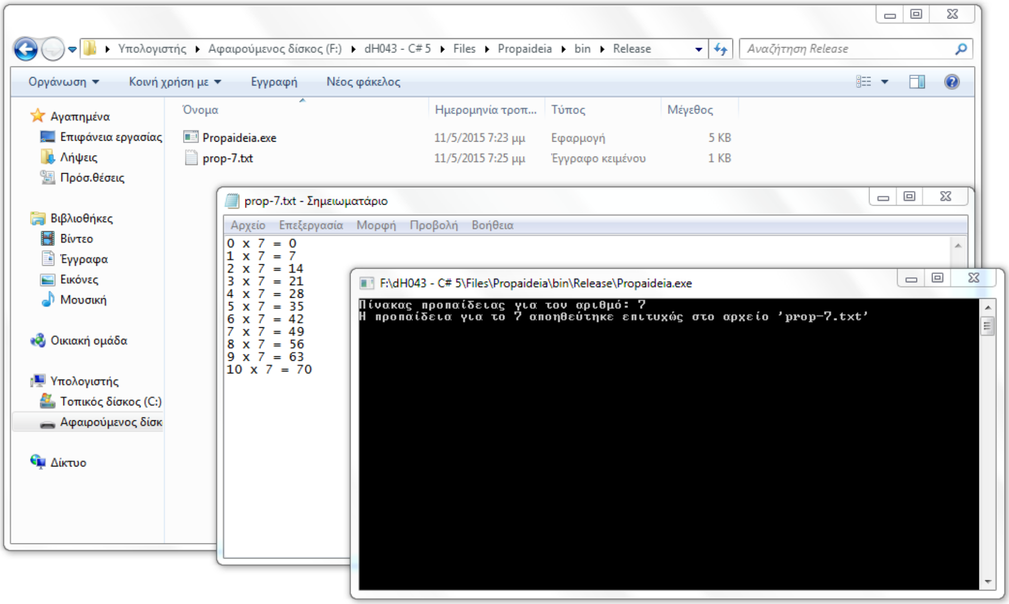
Η μελέτη μας θα ξεκινήσει από τις γραμμές 08 και 09. Εκεί λαμβάνουμε τα δεδομένα που εισάγει ο χρήστης και τα αποθηκεύουμε στη μεταβλητή usrin, ενώ ορίζουμε και τη μεταβλητή usrnum στην οποία θα αποθηκεύσουμε τον αντίστοιχο αριθμό. Η μετατροπή της συμβολοσειράς σε ακέραιο αριθμό συντελείται στο μπλοκ try (γραμμή 11), με τη βοήθεια της μεθόδου int.Parse(). Αν η συμβολοσειρά usrin δεν μπορεί να μετατραπεί σε ακέραιο αριθμό τότε εμφανίζουμε το μήνυμα της σχετικής εξαίρεσης, επιβάλλουμε μια παύση και στη συνέχεια τερματίζουμε τη λειτουργία του προγράμματος. Αν επιτευχθεί η μετατροπή, η ροή του προγράμματος μεταφέρεται πλέον στη γραμμή 18. Εκεί ορίζεται ένας πίνακας συμβολοσειρών, με 11 θέσεις. Όπως αντιλαμβάνεστε, σε αυτές τις θέσεις θα αποθηκευτούν τα αποτελέσματα από τον πολλαπλασιασμό των αριθμών 0 έως 10, με εκείνον που έδωσε ο χρήστης. Οι απαιτούμενες πράξεις και η συμπλήρωση του πίνακα πραγματοποιούνται στο βρόχο For (γραμμές 19 ως 2). Ο κορμός του προγράμματός μας (η μέθοδος main) ολοκληρώνεται με την αποθήκευση των περιεχομένων του πίνακα σε κάποιο αρχείο. Το όνομα του αρχείου συγκροτείται από το σταθερό τμήμα "prop-", τον αριθμό που έδωσε ο χρήστης και την κατάληξη ".txt" (γραμμή 24). Στη συνέχεια συναντάμε και πάλι ένα μπλοκ try, όπου γίνεται χρήση της μεθόδου WriteAllLines. Αν η εγγραφή του αρχείου ολοκληρωθεί χωρίς προβλήματα, εμφανίζεται το αντίστοιχο μήνυμα επιτυχίας (γραμμή 27). Σε διαφορετική περίπτωση εκτελείται το μπλοκ catch, που τυπώνει το σχετικό μήνυμα σφάλματος.

Νομίζουμε ότι ένα παράδειγμα με την WriteAllBytes θα ήταν περιττό. Όπως διαπιστώνετε, οι μέθοδοι για την εγγραφή δεδομένων σε αρχεία είναι εξαιρετικά απλές στη χρήση. Ας προχωρήσουμε τώρα και στην αντίστροφη διαδικασία.

Ανάγνωση δεδομένων

Προφανώς, για να διαβάσουμε το περιεχόμενο ενός αρχείου θα στραφούμε και πάλι στην κλάση File. Οι βασικές μέθοδοι που προσφέρονται γι' αυτή την εργασία είναι οι ακόλουθες:

- **File.ReadAllText(string path).** Αυτή η μέθοδος επιστρέφει ολόκληρο το περιεχόμενο του αρχείου σαν μια (ενιαία) συμβολοσειρά.



Το πρόγραμμά μας ετοίμασε ένα αρχείο με τον πίνακα προπαίδειας του 7. Δεν ξέρουμε για εσάς, αλλά το ξαδερφάκι που πηγαίνει δημοτικό ξετρελάθηκε :D

- **File.ReadAllLines(string path).** Κάθε γραμμή του αρχείου επιστρέφεται σαν μια ξεχωριστή συμβολοσειρά. Με αυτή τη μέθοδο μπορούμε να μεταφέρουμε όλες τις γραμμές ενός αρχείου σε έναν πίνακα.
- **File.ReadAllBytes(string path).** Η συγκεκριμένη μέθοδος προσφέρεται για αρχεία σαν αυτά που δημιουργούνται με τη μέθοδο WriteAllBytes. Η ReadAll Bytes επιστρέφει ένα προς ένα τα bytes που απαρτίζουν το περιεχόμενο του αρχείου.

Θα εξετάσουμε τώρα ένα απλό πρόγραμμα που ομοιάζει με το cat – το περίφημο εργαλείο που γνωρίζουμε από τη γραμμή εντολών των συστημάτων Unix. Το πρόγραμμά μας θα δέχεται σαν παράμετρο το όνομα του αρχείου που θέλουμε να τυπωθεί στην οθόνη. Εάν δεν του δώσουμε αυτή την παράμετρο, θα εμφανίζει ένα μήνυμα και θα περιμένει να πληκτρολογήσουμε κάποιο όνομα αρχείου.

```
using System;
using System.IO;

namespace FileContents {
 class Program {
    public static void Main( string[] args) {
      string fileName;
```

```
08       string fileContents;
09
10       if( args.Length > 0) {
11         fileName = args[0];
12       } else {
13         Console.Write( "Εμφάνιση περιεχομένων του: " );
14         fileName = Console.ReadLine();
15       }
16
17       if( File.Exists( filename ) ) {
18         try  {
19           fileContents = File.ReadAllText( fileName );
20           Console.WriteLine();
21           Console.WriteLine( fileContents );
22         }  catch ( Exception err ) {
23           Console.WriteLine( err.Message );
24         }
25       } else {
26         Console.WriteLine( "To αρχείο " + fileName + " δεν υπάρχει!" );
27       }
28
29       Console.ReadKey( true );
30     }
31   }
32 }
```

Στις γραμμές 07 και 08 ορίζουμε δύο μεταβλητές τύπου string, από τις οποίες η πρώτη θα περιέχει το όνομα του αρχείου και η δεύτερη το περιεχόμενό του. Στη γραμμή 10 συναντάμε για πρώτη φορά τον πίνακα args. Όπως μαντεύετε, ο εν λόγω πίνακας περιλαμβάνει τις παραμέτρους που δόθηκαν κατά την εκτέλεση του προγράμματος από τη γραμμή εντολών. Το πρόγραμμά μας ελέγχει το μήκος αυτού του πίνακα και, ουσιαστικά, τσεκάρει αν ο χρήστης έδωσε κάποια παράμετρο ή όχι. Αν έχουν δοθεί παράμετροι, θεωρούμε ότι η πρώτη από αυτές αντιστοιχεί στο όνομα του αρχείου που θέλει να τυπώσει ο χρήστης. Παρεμπιπτόντως, αν ο χρήστης θέλει να εμφανίσει ένα αρχείο με όνομα (ή διαδρομή) που περιλαμβάνει τον χαρακτήρα του κενού, θα πρέπει να περικλείσει την παράμετρο σε διπλά αγγλικά εισαγωγικά. Στην περίπτωση που δεν έχει οριστεί κάποια παράμετρος (γραμμές 12 έως 14), το πρόγραμμα εμφανίζει μια προτροπή και περιμένει από το χρήστη να εισάγει το όνομα κάποιου αρχείου.

Στη γραμμή 17 πραγματοποιείται ένα έλεγχος, για την ύπαρξη του αρχείου που προσδιόρισε ο χρήστης. Εάν δεν βρεθεί το αρχείο, η ροή του προγράμματος μεταφέρεται στη γραμμή 26. Εκεί εμφανίζεται ένα μήνυμα που ενημερώνει το χρήστη για την απουσία του αρχείου. Αν το αρχείο βρεθεί, ξεκινά η εκτέλεση ενός μπλοκ try.

Χρησιμοποιώντας τη μέθοδο File.ReadAllText, το περιεχόμενο του αρχείου μεταφέρεται ολόκληρο στη μεταβλητή fileContents. Αμέσως μετά, η τιμή της μεταβλητής τυπώνεται στην οθόνη. Όπως η εγγραφή, έτσι και η ανάγνωση ενός αρχείου μπορεί να αποτύχει για πολλούς λόγους. Για παράδειγμα, μπορεί ο χρήστης να στερείται των απαραίτητων δικαιωμάτων πρόσβασης, ή το μέγεθος του αρχείου να ξεπερνά τη διαθέσιμη μνήμη. Αν παρουσιαστεί κάποιο σφάλμα, η ροή του προγράμματος μεταφέρεται στο μπλοκ catch, από όπου και τυπώνεται το σχετικό μήνυμα λάθους.

Νομίζουμε ότι τα παραδείγματα που εξετάσαμε ως τώρα ήταν απλά και απολύτως κατανοητά. Στη συνέχεια θα μελετήσουμε ένα ελαφρώς πιο σύνθετο παράδειγμα. Το σχετικό πρόγραμμα θα μετράει τις γραμμές ενός αρχείου, θα εντοπίζει τις κενές και θα μπορεί να δημιουργεί αντίγραφα του δοθέντος αρχείου, με αριθμημένες γραμμές. Εν ολίγοις, θα μελετήσουμε ένα πρόγραμμα αντίστοιχο του lines.py, που μπορείτε να βρείτε στο http://www.multipetros.gr/?p=1924. Ας δούμε τώρα την υλοποίηση σε C#.

```
00 using System;
01 using System.IO;
02
03 namespace Lines  {
04   class Program  {
05     public static void Main( string[] args ) {
06       string fileName;
07       string[] fileLines;
08       int lines;
09       int emptyLines;
10
11       if( args.Length > 0 ) {
12         fileName = args[0];
13       } else  {
14         Console.Write( "Αρχείο: " );
15         fileName = Console.ReadLine();
16       }
17
18       if( File.Exists( fileName ) ) {
19         try {
20           fileLines = File.ReadAllLines( fileName );
```

```
      }  catch( Exception err ) {
        Console.WriteLine( err.Message );
        return;
      }
    } else {
      Console.WriteLine( "Το αρχείο " + fileName + " δεν υπάρχει!" );
      return;
    }

    lines = fileLines.Length;
    emptyLines = 0;

    int temp = lines;
    string format = "";
    while(  temp > 0 ) {
      format += "0";
      temp /= 10;
    }

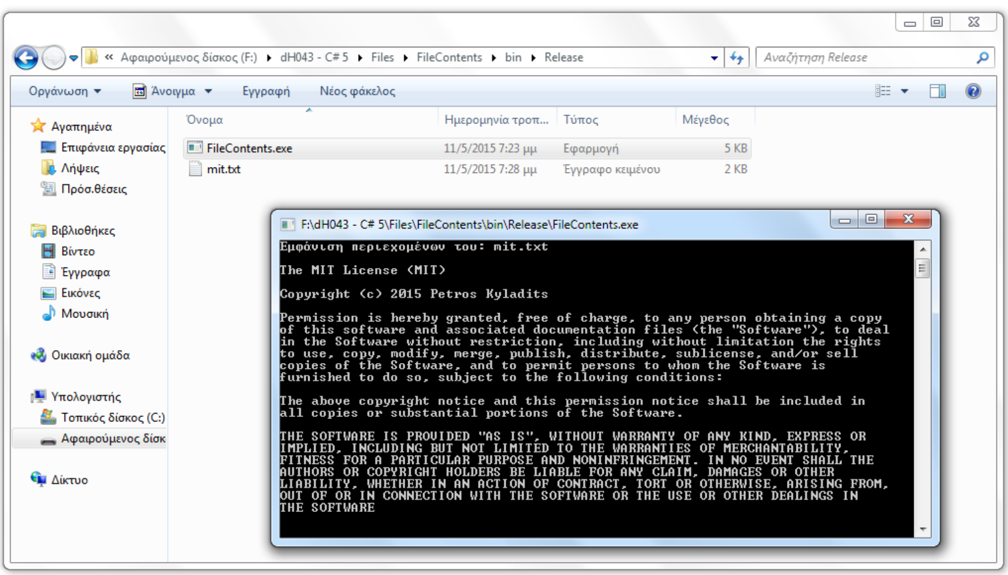
    for( int i=0; i<fileLines.Length; i++ ) {
      if( fileLines[i] == "" ) {
        emptyLines++;
      }
      fileLines[i] = i.ToString( format ) + " " + fileLines[i];
    }

    Console.WriteLine( "Στο αρχείο '" + fileName + "' βρέθηκαν " + lines + " γραμμές" );
    Console.WriteLine( "από τις οποίες οι " + emptyLines + " ήταν κενές.\n" );

    Console.Write( "Θέλετε να δημιουργηθεί νέο αρχείο με αριθμημένες\nτις γραμμές του '" + fileName + "'( Ναι, Οχι );" );
    string userAnswer = Console.ReadLine();
    if( userAnswer.StartsWith( "ν", StringComparison.CurrentCultureIgnoreCase ) || userAnswer.StartsWith( "n", StringComparison.CurrentCultureIgnoreCase ) ) {
      try {
        string newFileName = fileName+".lines-numbered";
```

```
55            File.WriteAllLines( newFileName, fileLines );
56            Console.WriteLine( "Δημιουργήθηκε επιτυχώς το '" + newFileName +
              "'" );
57          } catch( Exception err ) {
58            Console.WriteLine( err.Message );
59          }
60        }
61
62        Console.ReadKey( true );
63      }
64    }
65 }
```

Μέχρι και τη γραμμή 30 δεν συναντάμε μεγάλες διαφορές σε σχέση με το προηγούμενο πρόγραμμα. Μια αλλαγή που αξίζει να παρατηρήσουμε, είναι η αντικατάσταση της μεταβλητής fileContents από τον πίνακα fileLines (γραμμή 07). Αυτή τη φορά, βλέπετε, χρησιμοποιούμε τη μέθοδο ReadAllLines (γραμμή 20), η οποία επιστρέφει το περιεχόμενο του δοθέντος αρχείου γραμμή προς γραμμή. Μια ακόμα διαφοροποίηση σχετίζεται με την προσθήκη των μεταβλητών lines και emptyLines, οι οποίες θα συγκρατούν το πλήθος των γραμμών και το πλήθος των κενών γραμμών αντίστοιχα. Οι συγκεκριμένες μεταβλητές αρχικοποιούνται στις γραμμές 30 και 31.

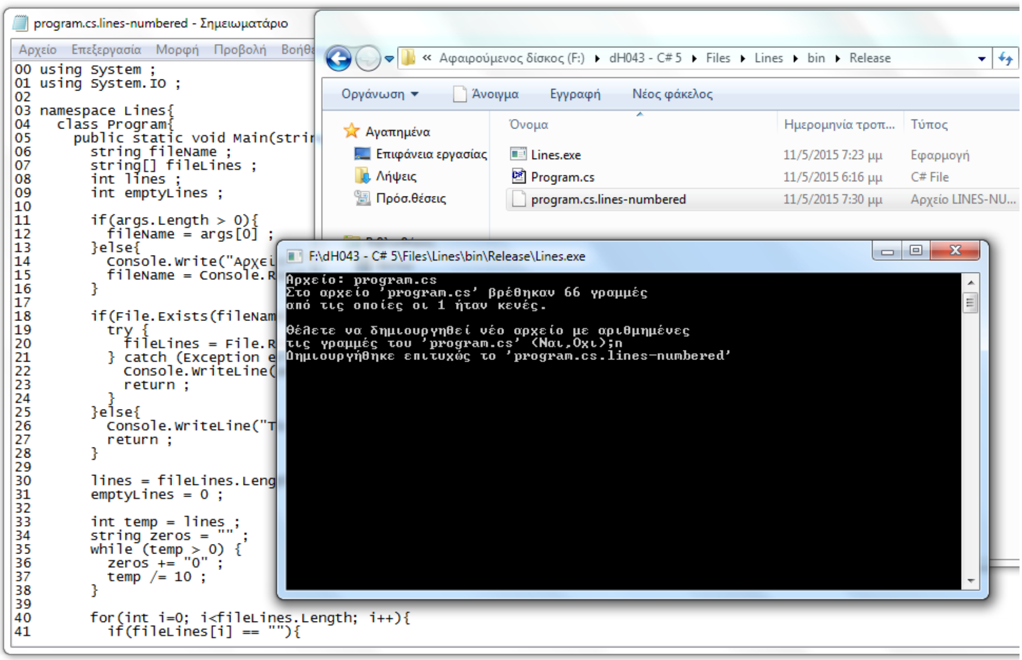


Η δική μας υλοποίηση του εργαλείου cat, για το .NET Framework. Εδώ βλέπουμε τα περιεχόμενα του αρχείου mit.txt (με την άδεια χρήσης MIT).

Στη συνέχεια πραγματοποιείται μια εργασία που θα σας φανεί τουλάχιστον παράξενη. Κατ' αρχάς, αντιγράφουμε το συνολικό πλήθος των γραμμών στη μεταβλητή temp. Στο βρόχο που ακολουθεί (γραμμές 35 έως 38) διαιρούμε τη μεταβλητή temp με τον αριθμό δέκα, έως ότου μηδενιστεί. (Σας κάνει εντύπωση το ότι κάποια στιγμή θα μηδενιστεί; Εφόσον πρόκειται για μεταβλητή τύπου Int, δεν διαθέτει δεκαδικό τμήμα. Έτσι, το αποτέλεσμα της διαίρεσης υπόκειται κάθε φορά σε μια βάρβαρη στρογγυλοποίηση.) Σε κάθε επανάληψη του βρόχου προσθέτουμε στη συμβολοσειρά format ένα μηδενικό. Με αυτόν τον τρόπο, η εν λόγω συμβολοσειρά αποκτά τόσα μηδενικά, όσα είναι και τα αριθμητικά ψηφία που εκφράζουν το συνολικό πλήθος των γραμμών του αρχείου. Η συμβολοσειρά format αξιοποιείται αργότερα (γραμμή 44), κατά το σχηματισμό των αριθμών γραμμής. Με τη βοήθειά της κάθε αριθμός θα έχει τόσα μηδενικά στα αριστερά του, ώστε όλοι να έχουν το ίδιο πλάτος! Με λίγα λόγια, η συμβολοσειρά format θα χρειαστεί για το λεγόμενο padding των αριθμών γραμμής. Ίσως να θεωρείτε περιττή όλη αυτή τη φασαρία, αλλά η δική μας τελειομανία επέβαλε την προσοχή και την επιμέλεια σε κάθε λεπτομέρεια – ακόμα και στο ζήτημα της μορφοποίησης των αριθμών γραμμής.

Ο βρόχος που ξεκινά στη γραμμή 40 πραγματοποιεί την καταμέτρηση των κενών γραμμών. Επιπρόσθετα, με τη βοήθεια της μεθόδου toString ο αριθμός της τρέχουσας γραμμής μετατρέπεται σε συμβολοσειρά και προστίθεται στην αρχή της γραμμής. Για την ώρα, όμως, οι ανανεωμένες γραμμές παραμένουν στον πίνακα fileLines και δεν αποθηκεύονται σε κάποιο αρχείο. Αμέσως μετά, το πρόγραμμα ενημερώνει το χρήστη για το συνολικό πλήθος των γραμμών του αρχείου, όπως και για το πλήθος των κενών. Τέλος, ζητάει την άδειά του για τη δημιουργία ενός αντιγράφου με αριθμημένες γραμμές. Η απόκριση του χρήστη συλλέγεται στη γραμμή 51 κι αποθηκεύεται στη μεταβλητή userAnswer. Ο έλεγχος της απάντησης πραγματοποιείται με τον πιο "ελαστικό" τρόπου που μπορούσαμε να φανταστούμε. Το πρόγραμμα δεν ελέγχει ούτε την ορθογραφία, ούτε και το πλήρες περιεχόμενο της απάντησης. Ο έλεγχος αφορά μόνο στο πρώτο γράμμα: Αν η απάντηση του χρήστη ξεκινά με το γράμμα "ν" ή το "n" (για τους φίλους που επιμένουν να γράφουν σε greeklish), θεωρούμε ότι ο χρήστης απάντησε καταφατικά. Στη γραμμή 52 συναντάμε τη μέθοδο StartsWith της κλάσης string, με την οποία υλοποιείται ο έλεγχος που περιγράψαμε. Η συγκεκριμένη μέθοδος δέχεται ως παραμέτρους τους χαρακτήρες που μας ενδιαφέρουν. Προαιρετικά, για κάθε χαρακτήρα δέχεται και μια πρόσθετη παράμετρο, που καθορίζει το πώς θα πραγματοποιηθεί η αναζήτηση. Αυτές οι βοηθητικές παράμετροι μπορούν να πάρουν μία από τις τιμές του enumeration ονόματι StringComparison (μπορείτε να δείτε όλα τα μέλη του συγκεκριμένου enumeration στο https://msdn.microsoft.com/en-us/library/system.stringcomparison). Εμείς επιλέξαμε το IgnoreCase, για να αποφύγουμε τη διάκριση μεταξύ πεζών και κεφαλαίων. Εφόσον το πρόγραμμα συμπεράνει ότι έλαβε θετική απάντηση, αποθηκεύει τα δεδομένα του πίνακα fileLines με τη βοήθεια της μεθόδου WriteAllLines (γραμμή 55). Κατά τα γνωστά, η εγγραφή του αρχείου είναι τοποθετημένη σε ένα μπλοκ try, ώστε μια ενδεχόμενη αποτυχία κατά την εγγραφή να μην επιφέρει τον τερματισμό του προγράμματος.

Τα παραδείγματα που εξετάσαμε μπορούν να δεχτούν πολλές προσθήκες, όπως και αρκετές βελτιώσεις. Μέχρι το επόμενο τεύχος θα σας προτείναμε να προσπαθήσετε να τα επεκτείνετε, με τις δικές σας ιδέες ή και με ιδέες που είδατε κάπου αλλού. Σε αυτό το στάδιο, εξάλλου, η συγγραφή προγραμμάτων και οι ατελείωτες δοκιμές έχουν μεγαλύτερη σημασία από το αν θα φτιάξετε κάτι πρωτότυπο ή όχι. Σημειώστε ότι μπορείτε να κατεβάσετε τον δικό μας κώδικα από το http://bit.ly/dH043dNETp5. Αν ανοίξετε το πακετάκι με τον κώδικα, θα διαπιστώσετε ότι έχουμε συμπεριλάβει και τα τέσσερα διαφορετικά projects που παρουσιάσαμε, κάτω από το ίδιο solution. Θα σας αφήσουμε να εξερευνήσετε αυτόν τον μυστηριώδη τρόπο οργάνωσης των αρχείων και είμαστε 102,59% σίγουροι, ότι μέχρι το επόμενο τεύχος θα έχετε λύσει το μυστήριο και θα έχετε κατανοήσει τη σχέση μεταξύ project και solution ;)



Το πρόγραμμα απαρίθμησης γραμμών απαριθμεί (duh!) τις γραμμές του αρχείου που περιλαμβάνει τον κώδικά του. Inception!

Skill: Beginner
Tags: SSH aliases, SSH tunnels, reverse SSH tunnels

Η φιλική πλευρά του SSH
[μέρος 1]

Όσοι ασχολούνται μανιωδώς με την ασφάλεια, αναπτύσσουν συγκεκριμένα και (επιτρέψτε μας) ιδιόρρυθμα αντανακλαστικά. Παράδειγμα: Κάθε φορά που ακούνε για το SSH, αναλογίζονται την κρυπτογραφική ισχύ των εμπλεκόμενων αλγορίθμων, συνυπολογίζουν τα πιο πρόσφατα σενάρια συνωμοσίας που άκουσαν και καταλήγουν σε μια εκτίμηση για την ασφάλεια που προσφέρουν οι σχετικές συνδέσεις. Για όλους τους υπόλοιπους, οι συνδέσεις SSH αποτελούν ένα εξαιρετικά χρήσιμο εργαλείο, που μπορεί να μας βοηθήσει σε αμέτρητες περιστάσεις.

του Παναγιώτη Βαρελά

Πριν από μερικούς μήνες αναφερθήκαμε στις θεωρητικές αδυναμίες που παρουσιάζουν ορισμένοι αλγόριθμοι, οι οποίοι εμπλέκονται στη δημιουργία των συνδέσεων SSH (βλ. "Ασφαλέστερο OpenSSH, εδώ και τώρα", στο τεύχος 40). Ξεκαθαρίσαμε βέβαια ότι πρόκειται για αδυναμίες που δεν πρέπει να μας προκαλούν ιδιαίτερη ανησυχία, αφού για την ώρα τουλάχιστον δεν υπάρχει κάποια γνωστή μέθοδος εκμετάλλευσής τους. Η παρουσίαση του θέματος είχε κυρίως προληπτικό χαρακτήρα και αφορούσε σε μια σειρά ρυθμίσεων που εξασφαλίζουν ότι οι συνδέσεις μας θα είναι όσο το δυνατόν πιο ασφαλείς στο κοντινό (ακριβέστερα: προβλέψιμο) μέλλον. Καλά όλα αυτά, θα πείτε, αλλά η δημιουργία και ο χειρισμός των συνδέσεων SSH δεν είναι πάντοτε εύκολη ή, τέλος πάντων, βολική εργασία. Μήπως θα μπορούσαμε να κάνουμε κάτι γι' αυτό το άμεσο πρόβλημα του σήμερα; Πώς θα μπορούσαμε να στήσουμε τη μία ή την άλλη σύνδεση, χωρίς κόπο; Με ποιους τρόπους θα μπορούσαμε να αξιοποιήσουμε μια υπάρχουσα σύνδεση;

Παρόμοια ερωτήματα απασχολούν κι εμάς. Έχουμε βλέπετε καμπόσα συστήματα που τρέχουν Linux, ορισμένα φυσικά και περισσότερα εικονικά, ενώ συντηρούμε και (τουλάχιστον) δύο VPSes. Έτσι, πολύ συχνά βρισκόμαστε στη γραμμή εντολών κάποιου συστήματος, προσπαθώντας να πάρουμε τις πιο πρόσφατες ενημερώσεις, να στήσουμε μια υπηρεσία ή εφαρμογή, να τροποποιήσουμε τις ρυθμίσεις κάποιας υπηρεσίας και πάει λέγοντας. Σ' όλες αυτές τις εργασίες οι συνδέσεις SSH αποδεικνύονται εξαιρετικά χρήσιμες, αφού διευκολύνουν τη "μετάβαση" από το ένα σύστημα

στο άλλο, τη μεταφορά αρχείων, την πραγματοποίηση δοκιμών κ.ο.κ. Επομένως, αν μας προβληματίζει κάτι σε σχέση με τη χρήση του SSH και μάλιστα σε καθημερινή βάση, αυτό αφορά στην ευκολία με την οποία κάνουμε τις δουλειές μας και όχι στο ενδεχόμενο να μας παρακολουθεί κάποιος πράκτορας. Σ' αυτό το άρθρο, λοιπόν, σας παρουσιάζουμε τους τρόπους με τους οποίους αξιοποιούμε *εμείς* τις συνδέσεις SSH. Θα ξεκινήσουμε από απλές ρυθμίσεις που διευκολύνουν την καθημερινή χρήση και θα φτάσουμε σε προχωρημένα κόλπα, για πιο εξεζητημένα και ενδιαφέροντα σενάρια χρήσης. Θέλουμε να πιστεύουμε ότι αυτά τα εφόδια θα σας λύσουν τα χέρια σε πολλές περιστάσεις, ενώ θα σας ενθαρρύνουν να πειραματιστείτε και να ανακαλύψετε ακόμα πιο ευρηματικές χρήσεις για τις συνδέσεις SSH.

Ταχύτητα και ευκολία

Αν και υποσχεθήκαμε να μην καταπιαστούμε με την ασφάλεια, είναι αδύνατο να μην πούμε ούτε μια κουβέντα. Η παρουσίασή μας άλλωστε θα ξεκινήσει με μια μέθοδο που απλοποιεί την εγκαθίδρυση συνδέσεων, ενώ ταυτόχρονα ενισχύει και την ασφάλεια του εκάστοτε SSH server. Πρόκειται για μια διαδικασία δύο απλών βημάτων, που αν την εφαρμόσουμε για κάθε μηχάνημα προς το οποίο συνδεόμαστε, η χρήση του ssh από τη γραμμή εντολών θα γίνει ευκολότερη και ταχύτερη από ποτέ.

Το πρώτο που πρέπει να κάνουμε είναι να απενεργοποιήσουμε τη χρήση των συνθηματικών και να επιβάλλουμε την ταυτοποίηση των χρηστών (user authentication) με κλειδιά κρυπτογράφησης. Με αυτόν τον τρόπο, οι SSH servers στους οποίους συνδεόμαστε δεν θα περιμένουν κάποιο συνθηματικό και ως εκ τούτου, θα είναι αδύνατο να υποστούν επιθέσεις brute force. Επιπρόσθετα, οι συνδέσεις μας θα πραγματοποιούνται χωρίς να απαιτείται η πληκτρολόγηση κάποιου συνθηματικού και χωρίς κάποια άλλη (αισθητή) καθυστέρηση. Επιγραμματικά, η διαδικασία για να ενεργοποιήσουμε αυτόν τον τρόπο σύνδεσης έχει ως εξής:

- Δημιουργούμε ένα ζεύγος δημόσιου-ιδιωτικού κλειδιού, στο μηχάνημα από το οποίο πραγματοποιούμε τις συνδέσεις.
- Εγκαθιστούμε το δημόσιο κλειδί στους επιθυμητούς λογαριασμούς χρήστη, των μηχανημάτων στα οποία συνδεόμαστε (SSH servers).
- Ρυθμίζουμε κατάλληλα τους SSH server, ώστε να μην περιμένουν συνθηματικό και να αξιοποιούν το δημόσιο κλειδί.

Η παραπάνω διαδικασία δεν παρουσιάζει ιδιαίτερη δυσκολία, αλλά χρειάζεται προσοχή. Παρ' όλα αυτά, καθώς πρόκειται για θέμα που έχουμε καλύψει πολλές φορές στο παρελθόν, δεν θα σας κουράσουμε επαναλαμβάνοντας τις λεπτομέρειες (βλ. "Ασφαλέστερο OpenSSH, εδώ και τώρα" και "Απομακρυσμένα ασφαλή logins, χωρίς password", στα τεύχη 040 και 042 αντίστοιχα).

Το δεύτερο βήμα για να πραγματοποιούμε συνδέσεις SSH με τη μέγιστη δυνατή άνεση, είναι να δημιουργήσουμε τα κατάλληλα SSH aliases στον υπολογιστή-πελάτη (υποθέτουμε ότι δουλεύετε από κάποιο Unix-like OS, όπως, π.χ., είναι οι διανομές Linux, τα BSD flavors ή το OS X). Πρόκειται για αυθαίρετα επιλεγμένες λέξεις στις οποίες έχουμε αντιστοιχίσει ένα σύνολο ρυθμίσεων. Για παράδειγμα, σε ένα SSH

alias μπορούμε να αντιστοιχίσουμε τη διεύθυνση ενός server και το όνομα χρήστη με το οποίο πραγματοποιούμε συνδέσεις προς τον συγκεκριμένο server. Τα SSH aliases ορίζονται μέσα στο σχετικό αρχείο ρυθμίσεων του λογαριασμού μας (~/.ssh/config) ή στο αρχείο ρυθμίσεων της υπηρεσίας του SSH (/etc/ssh/ssh_config). Προφανώς, στην πρώτη περίπτωση τα SSH aliases ισχύουν μόνο για το λογαριασμό μας, ενώ στη δεύτερη έχουν καθολική ισχύ. Πάντως για να αποφεύγουμε πιθανά μπερδέματα με άλλους χρήστες και για να είμαστε σίγουροι ότι τα aliases που έχουμε ορίσει δεν θα χαθούν εξαιτίας κάποιας αναβάθμισης του συστήματος, είναι προτιμότερο να τα τοποθετούμε στο αρχείο ρυθμίσεων του λογαριασμού μας (δηλαδή στο ~/.ssh/config).

Ο τρόπος δημιουργίας και χρήσης των SSH alias, όπως και η ευκολία που προσφέρουν, μπορεί να φανεί με τη βοήθεια ενός παραδείγματος. Ας υποθέσουμε ότι ο χρήστης spiral συνηθίζει να συνδέεται στο μηχάνημα deltahacker.gr, το οποίο δέχεται συνδέσεις SSH στη θύρα 45123. Ας υποθέσουμε, επίσης, ότι ο spiral συνδέεται στο εν λόγω μηχάνημα ως pvar. Για την πραγματοποίηση αυτής της σύνδεσης, κανονικά θα πρέπει να δώσει κάτι τέτοιο:

```
ssh pvar@deltahacker.gr -p 45123
```



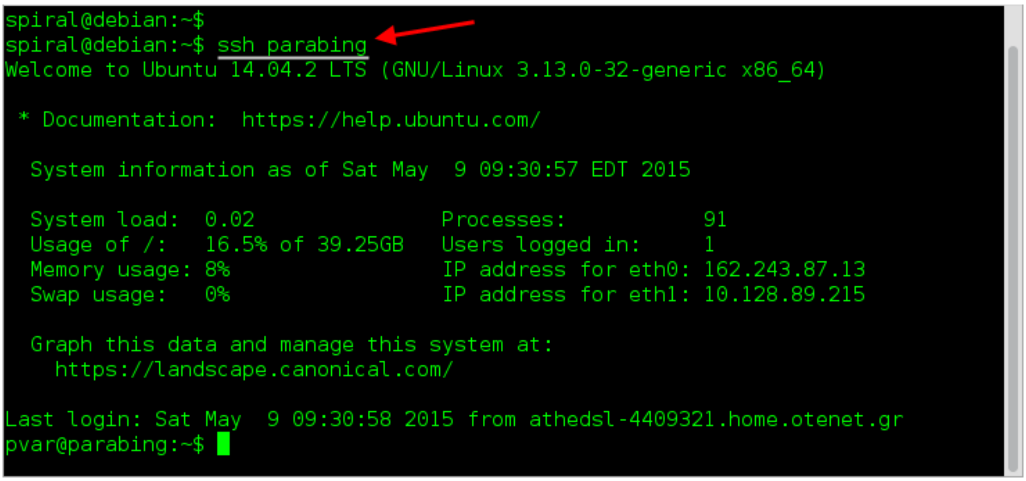
Τα SSH aliases θα αποτελούν "συντομεύσεις" προς συνδέσεις που πραγματοποιούμε συχνά. Ο ορισμός τους απαρτίζεται από μία λέξη που επιλέγουμε αυθαίρετα και από το επιθυμητό σύνολο ρυθμίσεων. Στην εικόνα βλέπουμε τον ορισμό του parabing, που αποτελεί ένα SSH alias για τη σύνδεση στο μηχάνημα parabing.com, στη θύρα 49250 και στο λογαριασμό pvar.

Αν ο spiral έχει δημιουργήσει ένα ζεύγος κλειδιών κι έχει εγκαταστήσει το δημόσιο κλειδί στον λογαριασμό του pvar στο deltahacker.gr, η σύνδεση θα πραγματοποιηθεί αμέσως και χωρίς καμία περαιτέρω καθυστέρηση. Όμως αυτή η διαδικασία μπορεί να απλοποιηθεί ακόμα περισσότερο, αρκεί ο spiral να ανοίξει το αρχείο ~/.ssh/config και να προσθέσει τα ακόλουθα:

```
Host delta
    HostName deltahacker.gr
    Port 45123
    User pvar
```

Με αυτόν τον τρόπο δημιουργείται ένα SSH alias (η λέξη "delta") που παραπέμπει στο deltahacker.gr, στο λογαριασμό του pvar και στη θύρα 45123. Μετά από αυτή τη μικρή προσθήκη στο αρχείο ρυθμίσεων, ο spiral θα μπορεί να συνδέεται εκτελώντας αυτό:

```
ssh delta
```



Έχοντας τοποθετήσει τον ορισμό του σχετικού SSH alias στο αρχείο ~/.etc/config, μπορούμε να το αξιοποιήσουμε ανά πάσα στιγμή και να συνδεθούμε με ευκολία στο μηχάνημα parabing.com.

Όπως βλέπετε, τα πράγματα είναι πολύ πιο εύκολα τώρα. Μη νομίζετε όμως ότι είμαστε τεμπέληδες. Η ευκολία δεν έγκειται στη λιγότερη πληκτρολόγηση. Σκεφτείτε το σενάριο κατά το οποίο θέλουμε να συνδεόμαστε σε δυο-τρία συστήματα, σε διαφορετικούς λογαριασμούς και σε διαφορετικές θύρες. Σε αυτή την περίπτωση, η χρήση των alias εξασφαλίζει ότι δεν θα μπερδέψουμε ποτέ τα μηχανήματα, τις θύρες και τους λογαριασμούς, ούτε θα χρειαστεί να αποστηθίσουμε τους σωστούς συνδυασμούς. Σκεφτείτε τώρα κι ένα script, που συνδέεται σε διάφορα μηχανήματα και πραγματοποιεί εργασίες όπως η λήψη ενημερώσεων, η περιοδική μεταβολή κάποιων ρυθμίσεων ασφαλείας και πάει λέγοντας. Με τα βήματα που περιγράψαμε, ένα τέτοιο script θα εκτελείται πάντοτε με επιτυχία, χωρίς να είμαστε μονίμως από πάνω του για να εισάγουμε κωδικούς πρόσβασης. Εξάλλου, η χρήση των SSH alias καθιστά τον κώδικα πιο σύντομο και περισσότερο ευανάγνωστο.

Εκτέλεση προγραμμάτων

Συχνά πυκνά προκύπτει η ανάγκη να τρέξουμε ένα ή περισσότερα προγράμματα σε κάποιο απομακρυσμένο μηχάνημα και είτε να παρακολουθήσουμε την έξοδό τους στην οθόνη, είτε να την αποθηκεύσουμε σε κάποιο αρχείο. Για παράδειγμα, μπορεί να θέλουμε να τρέξουμε το top σε κάποιο μικρό VPS, για να δούμε ποιο πρόγραμμα

το επιβαρύνει περισσότερο. Όταν θέλουμε να κάνουμε μια τέτοια δουλειά στα "πεταχτά", δεν είναι υποχρεωτικό να μεταφερθούμε στη γραμμή εντολών του ξένου συστήματος. Μπορούμε να κάνουμε τη δουλειά μας γρήγορα, χωρίς να εγκαταλείψουμε το δικό μας σύστημα. Συνεχίζοντας το παράδειγμα με το χρήστη spiral, αν αποφάσιζε να ρίξει μια ματιά στις ρυθμίσεις του nginx στο deltahacker.gr, θα μπορούσε να δώσει κάτι τέτοιο:

```
ssh delta cat /etc/nginx/sites-enabled/deltahacker
```

Το συγκεκριμένο κολπάκι είναι γνωστό σε αρκετούς. Υπάρχουν όμως κάποιες λεπτομέρειες γύρω από τη χρήση του, που δεν είναι το ίδιο διαδεδομένες.

Υπάρχει μια κατηγορία προγραμμάτων που εμφανίζουν χαρακτήρες σε "αυθαίρετες" θέσεις της οθόνης και όχι υποχρεωτικά σε διαδοχικές. Τέτοια προγράμματα είναι όσα σχεδιάζουν στην κονσόλα ένα περιβάλλον χαρακτήρων ή έστω μερικά στοιχεία ενός περιβάλλοντος, όπως, π.χ., είναι οι μπάρες προόδου, οι μπάρες κύλισης κ.ο.κ. Μερικά γνωστά προγράμματα της κατηγορίας είναι το top, το wget, το screen και όλοι οι text editors. Αρκετά από αυτά τα εργαλεία είναι αδύνατο να εκτελεστούν με τον τρόπο που δείξαμε παραπάνω, ενώ κάποια άλλα εκτελούνται, αλλά το περιβάλλον τους εμφανίζεται "σπασμένο". Για παράδειγμα, αν ο χρήστης spiral εκτελέσει το ακόλουθο:

```
ssh delta top
```

θα δει ένα μήνυμα λάθους και η εκτέλεση του top θα αποτύχει. Αντίθετα, αν δώσει κάτι τέτοιο:

```
ssh delta wget www.some-server.org/some-file.tar.bz
```

ο server του deltahacker θα αρχίσει να κατεβάζει κανονικά το αρχείο some-file.tar.bz, όμως η μπάρα προόδου του wget θα φαίνεται να μη λειτουργεί.

Για να μπορούν να σχηματίζουν το περιβάλλον τους και να λειτουργούν κανονικά, τα προγράμματα της κατηγορίας που περιγράφουμε πρέπει να εκτελούνται μέσα σε κάποιο τερματικό – πραγματικό (tty) ή εικονικό (pty). Οι συνδέσεις SSH, όταν μεταφερόμαστε στη γραμμή εντολών του απομακρυσμένου συστήματος, δημιουργούν ένα εικονικό τερματικό. Αν εκτελέσουμε το πρόγραμμα who, θα πληροφορηθούμε ότι βρισκόμαστε μέσα σε κάποιο pts (pseudo terminal slave: αποτελεί τμήμα/εξάρτημα ενός εικονικού τερματικού pty, διαβάστε περισσότερα πληκτρολογώντας man pty). Ωστόσο, όταν χρησιμοποιούμε το ssh για την εκτέλεση ενός προγράμματος *χωρίς* να μεταφερθούμε στη γραμμή εντολών του απομακρυσμένου συστήματος, δεν δημιουργείται κανενός είδους τερματικό. Όπως υποψιάζεστε, για να μπορούμε

να εκτελούμε οποιοδήποτε πρόγραμμα με το ssh και χωρίς να αντιμετωπίζουμε προβλήματα, αρκεί να επιβάλλουμε τη δημιουργία ενός εικονικού τερματικού. Κάτι τέτοιο γίνεται με τη χρήση της παραμέτρου –t. Αν ο χρήστης spiral πληκτρολογήσει

```
ssh -t delta top
```

Το πρόγραμμα top θα εκτελεστεί κανονικά. Παρεμπιπτόντως, η δημιουργία τερματικού θα μπορούσε να αυτοματοποιηθεί, με την κατάλληλη επέκταση στον ορισμό του SSH alias:

```
Host delta
    HostName deltahacker.gr

    Port 45123

    User pvar

    RequestTTY yes
```

Μια ακόμα ενδιαφέρουσα λεπτομέρεια σχετίζεται με τους τελεστές ανακατεύθυνσης. Ας υποθέσουμε ότι ο χρήστης spiral θέλει να αντιγράψει τις ρυθμίσεις του SSH server σε ένα αρχείο για λόγους backup. Έστω ότι εκτελεί το εξής:

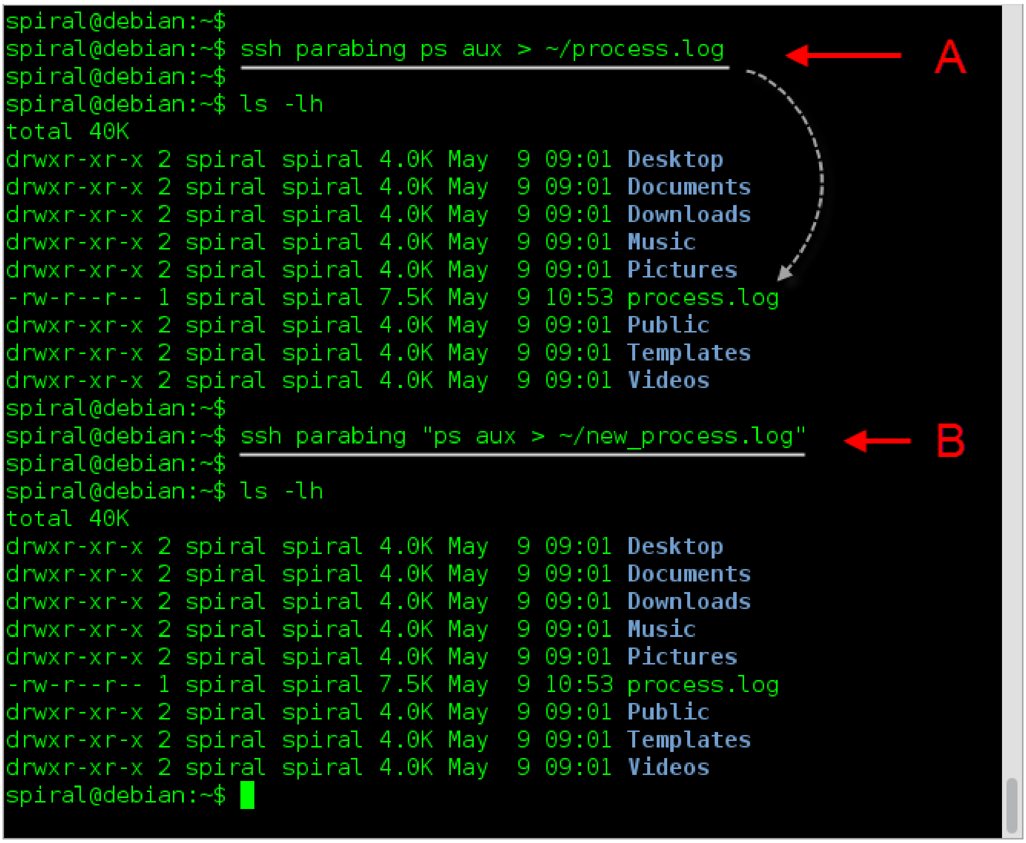
```
ssh delta cat /etc/ssh/sshd_config > ~/.ssh/server-config
```

Μπορείτε να φανταστείτε τι θα συμβεί; Οι ρυθμίσεις του SSH server του *απομακρυσμένου* συστήματος θα αποθηκευτούν στο αρχείο ~/.ssh/server-config του *τοπικού* συστήματος. Ο τελεστής ανακατεύθυνσης θα πάρει οτιδήποτε "έρχεται" από τα αριστερά του (ssh delta cat…) και θα το ρίξει στο αρχείο ~/.ssh/server-config. Αν το καλοσκεφτείτε, αυτή η συμπεριφορά είναι απόλυτα λογική και σε ορισμένες περιπτώσεις θα μπορούσε να είναι και επιθυμητή. Σκεφτείτε όμως το ενδεχόμενο να υπήρχε ήδη το αρχείο ~/.ssh/server-config στο τοπικό σύστημα. Σε αυτή την περίπτωση, ο spiral θα κατέστρεφε το τοπικό backup και μάλιστα χωρίς να το αντιληφθεί. Για να εκτελεστούν τα πάντα στο απομακρυσμένο σύστημα, ο spiral θα έπρεπε να περικλείσει την "εντολή" σε εισαγωγικά:

```
ssh delta "cat /etc/ssh/sshd_config > ~/.ssh/server-config"
```

Με αυτόν τον τρόπο, τόσο η εκτέλεση του cat όσο και η δημιουργία του αρχείου εξαιτίας του τελεστή ανακατεύθυνσης, θα πραγματοποιηθούν στον server του deltahacker. Όπως βλέπετε, όταν συνδυάζουμε τους τελεστές ανακατεύθυνσης με το ssh, πρέπει να επιλέγουμε προσεκτικά το αν θα χρησιμοποιήσουμε εισαγωγικά

ή όχι. Περιττό να πούμε ότι την ίδια προσοχή πρέπει να επιδεικνύουμε και στην περίπτωση που χρησιμοποιούμε τον τελεστή διασωλήνωσης (|) ή το χαρακτήρα διαχωρισμού εντολών (;). Χωρίς εισαγωγικά, το τμήμα της "εντολής" που βρίσκεται μετά τον ειδικό χαρακτήρα, εκτελείται τοπικά.



Η χρήση των τελεστών ανακατεύθυνσης χρειάζεται πάντοτε μεγάλη προσοχή. Στην πρώτη περίπτωση (χωρίς τα εισαγωγικά) το αρχείο processes.log δημιουργείται τοπικά. Στη δεύτερη περίπτωση (με τα εισαγωγικά) το αρχείο new_processes.log δημιουργείται στο απομακρυσμένο σύστημα.

Ασφαλής περιήγηση

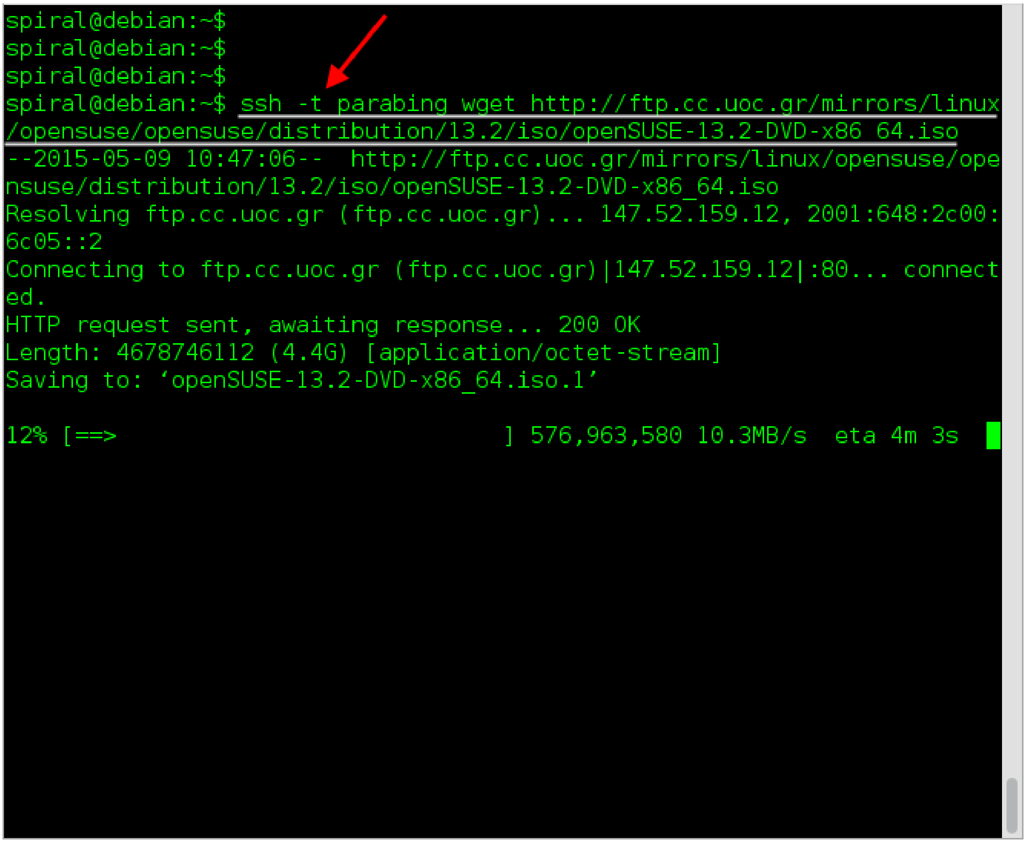
Όπως υποσχεθήκαμε, αυτή τη φορά δεν θα ασχοληθούμε με την ασφάλεια των συνδέσεων SSH. Αντίθετα, ξεκινώντας από αυτή θα μάθουμε να την εκμεταλλευόμαστε για την προστασία του web browsing! Αν πιστεύετε ότι μια τέτοια κίνηση θα αποτελούσε υπερβολή, σας συνιστούμε να το ξανασκεφτείτε. Φανταστείτε την περίπτωση κατά την οποία έχετε συνδεθεί με το laptop σε κάποιο πιθανά επικίνδυνο δίκτυο WiFi και θέλετε να διατηρήσετε ασφαλή τα δεδομένα που στέλνετε και λαμβάνετε από τον browser. Μήπως θεωρείτε ότι τα δίκτυα WiFi στα οποία συνδέεστε δεν κρύβουν κανέναν κίνδυνο; Αν και στο περιοδικό έχουμε παρουσιάσει

```
spiral@debian:~$
spiral@debian:~$ ssh parabing wget http://ftp.cc.uoc.gr/mirrors/linux/op
ensuse/opensuse/distribution/13.2/iso/openSUSE-13.2-DVD-x86_64.iso
--2015-05-09 10:39:26--  http://ftp.cc.uoc.gr/mirrors/linux/opensuse/ope
nsuse/distribution/13.2/iso/openSUSE-13.2-DVD-x86_64.iso
Resolving ftp.cc.uoc.gr (ftp.cc.uoc.gr)... 147.52.159.12, 2001:648:2c00:
6c05::2
Connecting to ftp.cc.uoc.gr (ftp.cc.uoc.gr)|147.52.159.12|:80... connect
ed.
HTTP request sent, awaiting response... 200 OK
Length: 4678746112 (4.4G) [application/octet-stream]
Saving to: ‘openSUSE-13.2-DVD-x86_64.iso’

     0K .......... .......... .......... .......... ..........  0%  180K
 7h3m
    50K .......... .......... .......... .......... ..........  0%  360K
 5h17m
   100K .......... .......... .......... .......... ..........  0% 2.06M
 3h43m
   150K .......... .......... .......... .......... ..........  0%  430K
 3h31m
   200K .......... .......... .......... .......... ..........  0% 2.20M
 2h56m
   250K .......... .......... .......... .......... ..........  0% 47.9M
 2h27m
   300K .......... .......... .......... .......... ..........  0%  431K
 2h31m
   350K .......... .......... .......... .......... ..........  0% 93.2M
 2h12m
   400K .......... .......... .......... .......... ..........  0% 2.16M
```

Όταν επιχειρούμε την εκτέλεση ενός προγράμματος σ' ένα απομακρυσμένο σύστημα, το ssh δεν θα δημιουργήσει τερματικό. Έτσι, τα προγράμματα που προϋποθέτουν την ύπαρξη τερματικού ενδέχεται να μην εκτελεστούν καθόλου – ή να παρουσιάσουν προβλήματα. Στην εικόνα βλέπουμε το wget να προσπαθεί μάταια να σχεδιάσει μια μπάρα προόδου.

πολλά θέματα που δείχνουν το ακριβώς αντίθετο, αν επιμένετε ξεχάστε τα περί ασφάλειας ή όχι των ασύρματων συνδέσεως και φανταστείτε το ακόλουθο σενάριο. Υποθέστε ότι βρισκόσαστε πίσω από ένα εξαιρετικά αυστηρό firewall, το οποίο απαγορεύει την πρόσβαση σε ορισμένες ιστοσελίδες. Μην τρέχει ο νους σας στο κινεζικό Great Firewall, αφού ένα αυστηρό firewall θα μπορούσε να βρίσκεται και στο εταιρικό δίκτυο κάποιου Δημόσιου Οργανισμού ή της δουλειάς σας. Νομίζουμε ότι σε μία τέτοια περίπτωση θα επιθυμούσατε διακαώς (έστω κι από πείσμα) να ξεπεράσετε τους περιορισμούς του firewall. Αν εξακολουθείτε να πιστεύετε ότι γινόμαστε παρανοϊκοί, σκεφτείτε την απλούστερη περίπτωση όπου θέλετε να συνδεθείτε σε μια υπηρεσία streaming σαν αυτή του Netflix ή του LastFM. Ως γνωστόν, τέτοιες υπηρεσίες προσφέρονται μόνο για τους κατοίκους των ΗΠΑ και ορισμένων χωρών της Ευρώπης. Ε, λοιπόν, αξιοποιώντας μια σύνδεση SSH, μπορούμε να ξεγελάσουμε οποιαδήποτε υπηρεσία του είδους και να εμφανιστούμε ως βέροι Νεοϋορκέζοι, γνήσιοι Ισλανδοί ή δεν ξέρω κι εγώ τι άλλο.

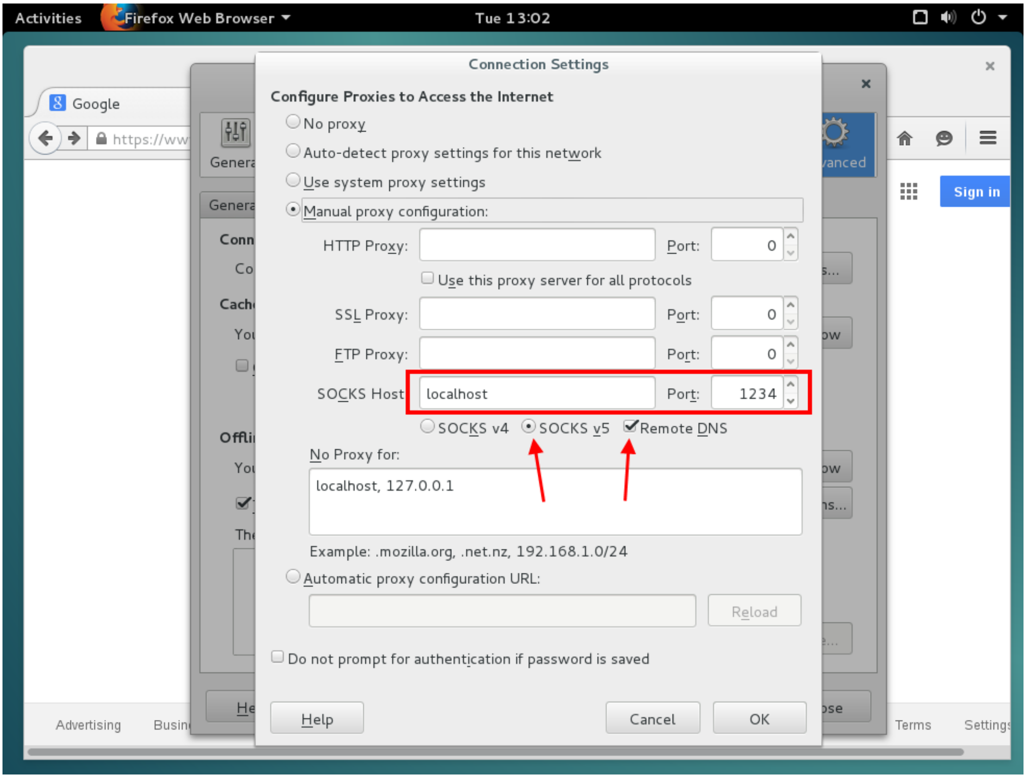


Με τη βοήθεια της παραμέτρου –t ζητήσαμε από το ssh να δημιουργήσει ένα τερματικό. Τώρα το wget λειτουργεί κανονικά.

Το OpenSSH, η υλοποίηση του SSH που χρησιμοποιούμε πρακτικά όλοι, έχει τη δυνατότητα να δημιουργεί ένα ιδιαίτερο είδος συνδέσεων, που ονομάζονται SSH tunnels. Αυτές οι συνδέσεις είναι πλήρως κρυπτογραφημένες –όπως άλλωστε όλες οι συνδέσεις SSH–, ενώ παρέχουν και μια πρόσθετη δυνατότητα: Λειτουργούν ως σήραγγες, μέσα από τις οποίες μπορούν να διοχετευτούν άλλες συνδέσεις, οποιουδήποτε πρωτοκόλλου. Για να πετύχουμε όσα περιγράψαμε παραπάνω, αρκεί να διοχετεύσουμε τις συνδέσεις που δημιουργεί ο browser μέσα από ένα SSH tunnel. Μη νομίζετε ότι κάτι τέτοιο απαιτεί πολλά και περίπλοκα βήματα. Πρόκειται για μία από τις σπάνιες περιπτώσεις, που η υλοποίηση είναι εξίσου απλή με την περιγραφή :) Αρκεί να δώσουμε κάτι τέτοιο:

```
ssh -fNTD 1234 username@some-server.org
```

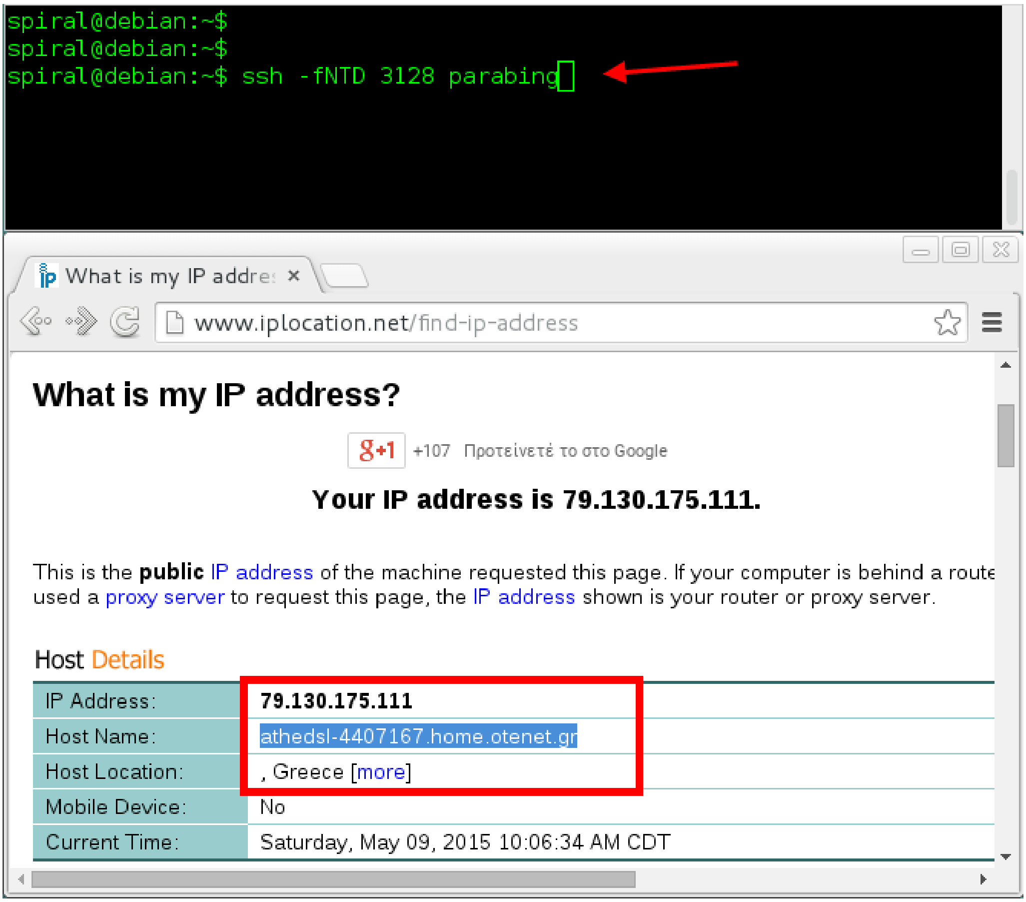
Με την εκτέλεση του παραπάνω θα δημιουργηθεί μια "σήραγγα SSH", που θα έχει ως είσοδο τη θύρα 1234 του τοπικού συστήματος και σαν έξοδο το μηχάνημα some-server.org. Αν εξαιρέσουμε το γεγονός ότι η έξοδος του tunnel χαρακτηρίζεται ως



Οι απαραίτητες ρυθμίσεις στον Firefox, για την αξιοποίηση του SSH tunnel που στήσαμε. Για περισσότερη ασφάλεια, μαζί με την προώθηση των αιτημάτων HTTP πρέπει να επιβάλλουμε και την προώθηση των αιτημάτων DNS.

Dynamic και δεν είναι προσαρτημένη σε κάποια συγκεκριμένη θύρα, η όλη λειτουργία του tunnel θυμίζει το port forwarding των router. Με τις παραμέτρους που δώσαμε στο ssh, η σύνδεσή μας θα μεταφερθεί αυτόματα στο υπόβαθρο και δεν θα δεσμεύσει την κονσόλα (παράμετρος –f), ούτε θα δημιουργήσει κάποιο εικονικό τερματικό (παράμετρος –T). Η προώθηση των συνδέσεων (forwarding) πραγματοποιείται με την παράμετρο –D, ενώ η παράμετρος –N εξασφαλίζει ότι δεν θα εκτελεστεί καμία εντολή στο απομακρυσμένο σύστημα.

Έχοντας έτοιμο το SSH tunnel, απομένει η κατάλληλη ρύθμιση του browser. Στην περίπτωση του Firefox, αρκεί να επισκεφθούμε τις ρυθμίσεις, να μεταβούμε στην ενότητα Advanced και από εκεί στην καρτέλα Network. Στην ενότητα Connection του παραθύρου που εμφανίζεται, θα πατήσουμε το κουμπί Settings. Με αυτόν τον τρόπο βρίσκουμε τις ρυθμίσεις που αφορούν στη χρήση κάποιου proxy server. Εκεί επιλέγουμε το Manual Proxy configuration και τσεκάρουμε τις επιλογές SOCKS v5 και Remote DNS. Φυσικά, πριν πατήσουμε στο ΟΚ πρέπει να συμπληρώσουμε και τα κατάλληλα πεδία, δίπλα στην ετικέτα SOCKS host. Από εκεί προσδιορίζουμε το σημείο εισόδου του tunnel που δημιουργήσαμε προηγουμένως (διεύθυνση localhost

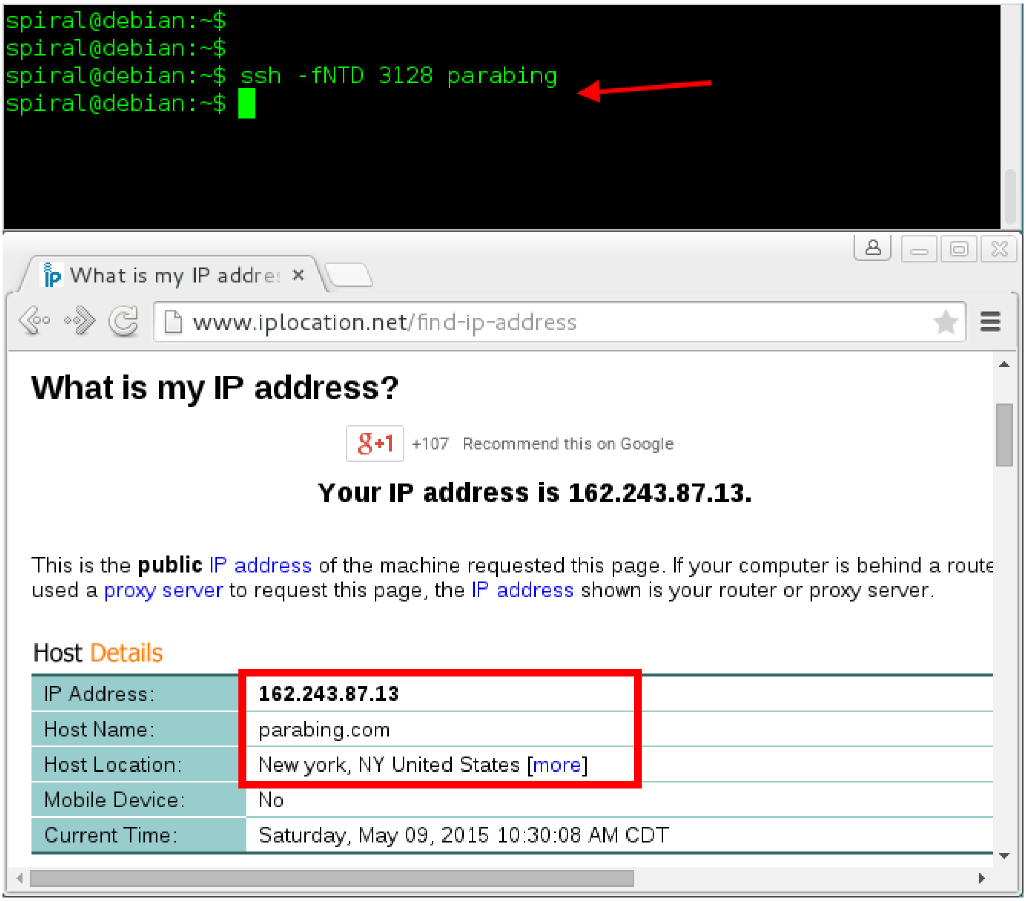


Λίγο πριν στήσουμε το SSH tunnel, μια επίσκεψη στην υπηρεσία iplocation δείχνει ότι βρισκόμαστε κάπου στην Ελλάδα.

και θύρα 1234). Στην περίπτωση του Chrome, η σχετική ρύθμιση επιτυγχάνεται με τη βοήθεια παραμέτρων που πρέπει να δώσουμε κατά την εκκίνηση του browser. Για την ακρίβεια, πρέπει να τον ξεκινήσουμε κάπως έτσι:

```
google-chrome --proxy-server="socks5://localhost:1234" --host-resolver-
rules="MAP * 0.0.0.0, EXCLUDE localhost" &
```

Αφού ξεμπερδέψουμε με τις ρυθμίσεις του browser, οι περιπλανήσεις μας στο web θα είναι πλήρως κρυπτογραφημένες και προφυλαγμένες από τα αδιάκριτα μάτια κάθε περίεργου (προσοχή: η προστασία αφορά *μόνο* στις συνδέσεις που δημιουργεί ο browser). Επιπρόσθετα, ακόμα κι αν βρισκόμαστε πίσω από κάποιο εξαιρετικά αυστηρό Firewall, θα μπορούμε να συνδεόμαστε οπουδήποτε. Για την ακρίβεια, θα μπορούμε να συνδεθούμε οπουδήποτε μπορεί να συνδεθεί το μηχάνημα που απο-



Αφού στήσαμε το SSH tunnel και ρυθμίσαμε κατάλληλα τον browser, επαναλάβαμε την επίσκεψη στην υπηρεσία iplocation. Αυτή τη φορά, ο υπολογιστής μας έμοιαζε να βρίσκεται στη Νέα Υόρκη. Βλέπετε, ο server που λειτουργεί ως "έξοδος" για το SSH tunnel βρίσκεται στην άλλη όχθη του Ατλαντικού.

τελεί την έξοδο του tunnel. Το αυστηρό firewall που έχουμε μπροστά μας θα βλέπει μόνο μια εξερχόμενη σύνδεση SSH και τίποτα άλλο. Τέλος, για το υπόλοιπο Internet, οι συνδέσεις μας θα μοιάζουν να προέρχονται από τον τόπο κατοικίας του some-server.org (η έξοδος του tunnel). Αυτό σημαίνει ότι αν ο συγκεκριμένος server βρίσκεται στην Αμερική, θα μπορούμε να συνδεθούμε και στο Netflix και στην υπηρεσία streaming του LastFM (εκτός κι αν η IP του server είναι σε κάποια blacklist – αλλά δεν χρειάζεται να είμαστε απαισιόδοξοι).

Πριν προχωρήσουμε, αξίζει να σταθούμε σε μια ενδιαφέρουσα λεπτομέρεια. Περιγράφοντας τη ρύθμιση του Firefox, αναφέραμε ότι πρέπει να ενεργοποιήσουμε την επιλογή Remote DNS. Με αυτόν τον τρόπο εξασφαλίζουμε την προώθηση των αιτημάτων DNS (από πλευράς browser) μέσα από το SSH tunnel και κατ' επέκταση την κρυπτογράφησή τους. Αναρωτιέστε γιατί μας ενδιαφέρει κάτι τέτοιο; Η αλήθεια είναι ότι η συγκεκριμένη ρύθμιση έχει σημασία μόνον όταν βρισκόμαστε σε

άγνωστα και πιθανώς εχθρικά –ή έστω αυστηρά– τοπικά δίκτυα. Αν δεν προφυλάσσουμε τα αιτήματα DNS, οποιοσδήποτε παρακολουθεί το τοπικό δίκτυο θα μαθαίνει αμέσως με ποιους servers μιλάμε. Μπορεί να μη βλέπει τα δεδομένα που στέλνουμε και λαμβάνουμε –αυτά κρυπτογραφούνται από το SSH tunnel–, αλλά θα ξέρει που πηγαίνουμε και με ποιους μιλάμε. Επιπρόσθετα, θα είναι ενδεχομένως σε θέση να πραγματοποιήσει επιθέσεις DNS και να μας παραπέμψει ύπουλα σε δικούς τους, κακόβουλους servers. Σημειώστε ότι ο Chrome δεν διαθέτει κάποια ρύθμιση, αντίστοιχη με αυτή του Firefox. Ωστόσο, φροντίζει για την αυτόματη προώθηση των αιτημάτων DNS μέσα από το ίδιο "κανάλι" που ταξιδεύουν και τα αιτήματα HTTP. Αν δεν είσαστε σίγουροι για τη συμπεριφορά τους δικού σας browser ή αν απλά θέλετε να ελέγξετε τις ρυθμίσεις σας, αρκεί να παρακολουθήσετε τις συνδέσεις που εξέρχονται από το SSH tunnel. Για παράδειγμα, αν θέλετε να τσεκάρετε ότι όντως προωθούνται τα αιτήματα DNS, μπορείτε να αναζητήσετε συνδέσεις που έχουν στον προορισμό τους τη θύρα 53 (πρόκειται για τη θύρα που χρησιμοποιείται εξ ορισμού από τους DNS servers) του απομακρυσμένου SSH server. Κάτι τέτοιο μπορεί να γίνει ως εξής:

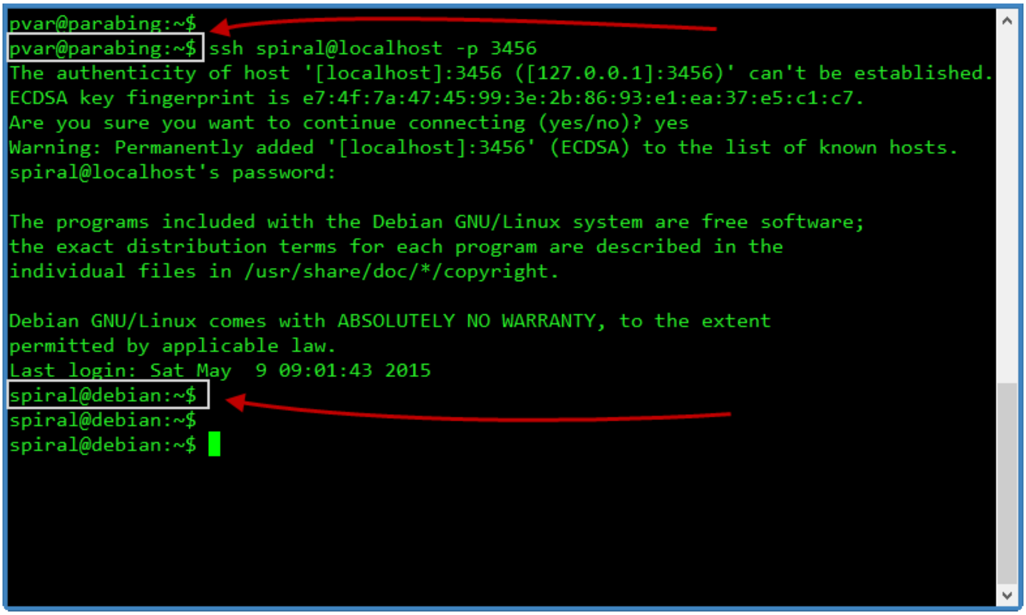
```
ssh username@some-server.org tcpdump -i eth0 port 53
```

```
admin@parabing:/home/pvar$
admin@parabing:/home/pvar$
admin@parabing:/home/pvar$ sudo tcpdump -i eth0 port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
12:18:53.514255 IP parabing.com.46439 > google-public-dns-b.google.com.domain:
 4166+ A? fbcdn-dragon-a.akamaihd.net. (45)
12:18:53.514276 IP parabing.com.46439 > google-public-dns-b.google.com.domain:
 60543+ AAAA? fbcdn-dragon-a.akamaihd.net. (45)
12:18:53.540545 IP google-public-dns-b.google.com.domain > parabing.com.46439:
 4166 4/0/0 CNAME fbcdn-dragon-a.akamaihd.net.edgesuite.net., CNAME a1170.dsw4
.akamai.net., A 23.62.7.17, A 23.62.7.65 (161)
12:18:53.543388 IP google-public-dns-b.google.com.domain > parabing.com.46439:
 60543 4/0/0 CNAME fbcdn-dragon-a.akamaihd.net.edgesuite.net., CNAME a1170.dsw
4.akamai.net., AAAA 2600:1408:17::173e:653, AAAA 2600:1408:17::173e:650 (185)
12:18:54.435439 IP parabing.com.51856 > google-public-dns-b.google.com.domain:
 23560+ PTR? 4.4.8.8.in-addr.arpa. (38)
12:18:54.453800 IP google-public-dns-b.google.com.domain > parabing.com.51856:
 23560 1/0/0 PTR google-public-dns-b.google.com. (82)
12:18:54.454070 IP parabing.com.51874 > google-public-dns-b.google.com.domain:
```

Χρησιμοποιώντας το tcpdump, επιβεβαιώσαμε ότι τα αιτήματα DNS προωθούνται μέσω του SSH tunnel. Οποιοσδήποτε μέσα στο ίδιο τοπικό δίκτυο προσπαθεί να παρακολουθήσει τις κινήσεις μας, θα μείνει στο σκοτάδι.

Το αντίστροφο πρόβλημα

Με το τέχνασμα που εξετάσαμε μπορούμε να αποδράσουμε από ένα περιοριστικό firewall – ή να ξεγλιστρήσουμε από την παρακολούθηση και τις παγίδες κάποιου κακόβουλου χρήστη ή μανιακού διαχειριστή. Σε κάθε περίπτωση, με τη βοήθεια ενός SSH tunnel μπορούμε να *εξέλθουμε* από ένα τοπικό δίκτυο, με ευκολία και ασφά-



Παρατηρείστε ότι αρχικά βρισκόμαστε στο μηχάνημα parabing. Από εκεί, επιχειρώντας μια σύνδεση στο ίδιο μηχάνημα και στη θύρα 3456, καταλήγουμε στο (εικονικό) μηχάνημα ονόματι debian, που βρίσκεται στο οικιακό μας δίκτυο. Ουσιαστικά, μόλις αξιοποιήσαμε ένα reverse SSH tunnel που είχαμε στήσει νωρίτερα.

λεια. Τι γίνεται όμως όταν θέλουμε να *εισέλθουμε* σε κάποιο τοπικό δίκτυο; Μη νομίζετε ότι το ρίξαμε ξαφνικά στις επιθέσεις. Μιλάμε για τη σύνδεση σε κάποιο οικείο τοπικό δίκτυο, που για διάφορους τεχνικούς λόγους μπορεί να είναι από δύσκολη έως αδύνατη. Ας υποθέσουμε ότι θέλουμε να συνδεθούμε σε έναν υπολογιστή του οικιακού μας δικτύου, ενώ βρισκόμαστε κάπου έξω. Η πραγματοποίηση αυτής της σύνδεσης είναι αρκετά μπελαλίδικη: Πρέπει να έχουμε ανοίξει κάποιο port στο firewall του οικιακού μας router και να έχουμε πραγματοποιήσει το κατάλληλο port forwarding, προς το μηχάνημα που μας ενδιαφέρει. Επιπρόσθετα, πρέπει να γνωρίζουμε και την τρέχουσα διεύθυνση IP που έχει αντιστοιχιστεί στην οικιακή μας σύνδεση. Το πρόβλημα με αυτή τη λύση δεν είναι μόνο ότι απαιτεί πολλές ρυθμίσεις. Θα τη χαρακτηρίζαμε άκαμπτη, ενώ σε ορισμένες περιπτώσεις είναι και εντελώς άχρηστη, αφού δεν μπορούμε να την εφαρμόσουμε. Για παράδειγμα, σκεφτείτε την περίπτωση που θέλουμε να συνδεθούμε σε κάποιο μηχάνημα της δουλειάς ή της σχολής, από το σπίτι. Κανένας λογικός διαχειριστής δεν θα δεχτεί να αλλάξει τις ρυθμίσεις του firewall της δουλειάς ή της σχολής, μόνο και μόνο για τη δική μας διευκόλυνση. (Εκτός κι αν ο διαχειριστής είμαστε εμείς. Πάντως ακόμη και τότε υπάρχουν πολλοί λόγοι για να μη σκαλίσουμε το firewall.)

Όπως υποψιάζεστε, οι συνδέσεις SSH μπορούν και πάλι να δώσουν τη λύση. Αυτή τη φορά θέλουμε να μπούμε και όχι να βγούμε από κάποιο τοπικό δίκτυο. Επομένως, χρειαζόμαστε το ακριβώς αντίθετο των SSH tunnel: ένα reverse SSH tunnel. Αυτές οι συνδέσεις συμπεριφέρονται και πάλι ως σήραγγες, μόνο που επιτρέπουν την πραγματοποίηση συνδέσεων με την αντίθετη φορά. Με άλλα λόγια, η είσοδός τους

βρίσκεται στο απομακρυσμένο σύστημα και η έξοδος στο τοπικό. Για να λύσουμε το πρόβλημα που περιγράψαμε αρχικά, αρκεί να δημιουργήσουμε ένα reverse SSH tunnel από το μηχάνημα του σπιτιού προς κάποιο VPS. Αργότερα, όταν θελήσουμε να συνδεθούμε στο μηχάνημα του σπιτιού, αρκεί να συνδεθούμε στον VPS και να αξιοποιήσουμε το "αντεστραμμένο τούνελ". Φυσικά, στη θέση του VPS θα μπορούσε να είναι οποιοδήποτε άλλο μηχάνημα είναι εύκολα προσβάσιμο από το Internet. Για παράδειγμα, κάποιος server του δικτύου της δουλειάς ή της σχολής. Το στήσιμο ενός reverse SSH tunnel είναι πανεύκολη υπόθεση. Συνεχίζοντας το προηγούμενο σενάριο, αρκεί να ανοίξουμε τη γραμμή εντολών του οικιακού μηχανήματος και να δώσουμε κάτι τέτοιο:

```
ssh -fNTR 2222:localhost:22 username@some-server.org
```

Με αυτόν τον τρόπο δημιουργείται ένα αντεστραμμένο τούνελ, που έχει σαν είσοδο τη θύρα 2222 του μηχανήματος some-server.org και σαν έξοδο τη θύρα 22 του τοπικού συστήματος. Όπως είδαμε και νωρίτερα, οι παράμετροι –f, -T και –N εξασφαλίζουν ότι η συγκεκριμένη σύνδεση θα τοποθετηθεί στο background, δεν θα δημιουργήσει τερματικό και δεν θα εκτελέσει εντολές στο απομακρυσμένο μηχάνημα. Η δημιουργία του reverse SSH tunnel εξασφαλίζεται από την παράμετρο –R. Εφόσον ετοιμάσουμε το τούνελ, δεν χρειάζεται να κάνουμε κάτι άλλο. Όταν θελήσουμε να συνδεθούμε στον υπολογιστή του οικιακού δικτύου από κάπου αλλού, αρκεί να συνδεθούμε στο μηχάνημα some-server.org και να δώσουμε κάτι τέτοιο:

```
ssh localhost -p 2222
```

Με την παραπάνω γραμμή επιχειρούμε μια σύνδεση SSH προς τη θύρα 2222 του τοπικού συστήματος, που στη συγκεκριμένη περίπτωση είναι ο some-server.org. Ωστόσο, εξαιτίας του reverse SSH tunnel που δημιουργήσαμε νωρίτερα, η συγκεκριμένη σύνδεση θα προωθηθεί στη θύρα 22 του μηχανήματος στο σπίτι.

Ελπίζουμε όλες αυτές οι περιγραφές των συνδέσεων, των αντιστρόφων και των ευθειών, να μην σας προκάλεσαν πονοκέφαλο. Όπως και να 'χει, μην ξεχνάτε ότι ο υπέρτατος κανόνας της μάθησης ισχύει κι εδώ: Για να αφομοιώσετε όσα παρουσιάσαμε, μαζί με τη θεωρητική μελέτη θα χρειαστεί και η πρακτική εξάσκηση. Αν δεν έχετε το απαιτούμενο πλήθος μηχανημάτων, σας προτείνουμε να δημιουργήσετε μερικά εικονικά συστήματα και να αρχίσετε τις δοκιμές από σήμερα. Στο επόμενο τεύχος θα παρουσιάσουμε μερικά πιο προχωρημένα κολπάκια και θα πρέπει να είσαστε έτοιμοι.

Skill: Intermediate
Tags: mailserver, domain name, WOIS protection, forward DNS resolution, reverse DNS resolution, reverse DNS, PTR record, permanent resize, flexible resize, security hardening

Υποδομή για τον mailserver μας

Αποφασίσατε λοιπόν να δημιουργήσετε τον δικό σας mailserver. Θα αποκτήσετε έτσι περισσότερο έλεγχο της ηλεκτρονικής σας αλληλογραφίας και, σιγά σιγά, θα απαγκιστρωθείτε από τους κυρίαρχους mail providers. Δεχθείτε παρακαλούμε τα συγχαρητήριά μας γι' αυτή σας την απόφαση – και πάμε να δούμε πώς θα βάλουμε τις σωστές βάσεις για το φιλόδοξο αυτό project.

του Χρήστου Βαρελά

Σε αυτό το μέρος του αφιερώματός μας απασχολούμαστε με τις ακόλουθες εργασίες:

- Λήψη domain από το Gandi.net
- Επιλογή και αρχική ρύθμιση του κατάλληλου VPS, από τη DigitalOcean
- Βασικές ρυθμίσεις του VPS, μετά την πρώτη εκκίνησή του
- Απολύτως απαραίτητες εργασίες υπέρ της ασφάλειας

Σε κάθε στάδιο παρουσιάζουμε τη λογική των επιλογών/κινήσεών μας, ωστόσο θα θέλαμε να υπογραμμίσουμε από τώρα ότι δεν είναι υποχρεωτικό να πάρετε domain από το Gandi, ούτε VPS από τη DigitalOcean. Και μετά από αυτό, ας πιάσουμε δουλειά.

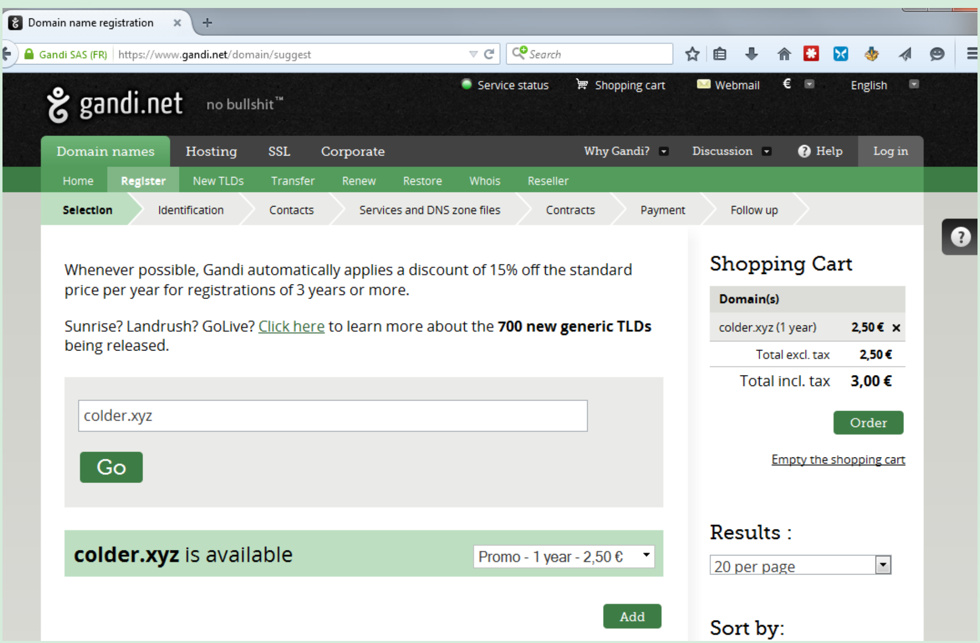
Απόκτηση νέου domain, για το νέο email

Κάθε mailserver εξυπηρετεί ένα domain, γι' αυτό και λέμε "ο mailserver για το domain". Όπως αναφέραμε και στο πρώτο μέρος του αφιερώματός μας, που αρχίζει από τη σελίδα 14, δεν θέλαμε να πειράξουμε τα (Google Apps hosted) emails για τα domains που χρησιμοποιούμε εδώ και χρόνια (defiant.gr, deltahacker.gr και parabing.com). Έτσι, προκειμένου ο mailserver μας να εξυπηρετεί κάποιο domain, αγοράσαμε άλλο ένα.

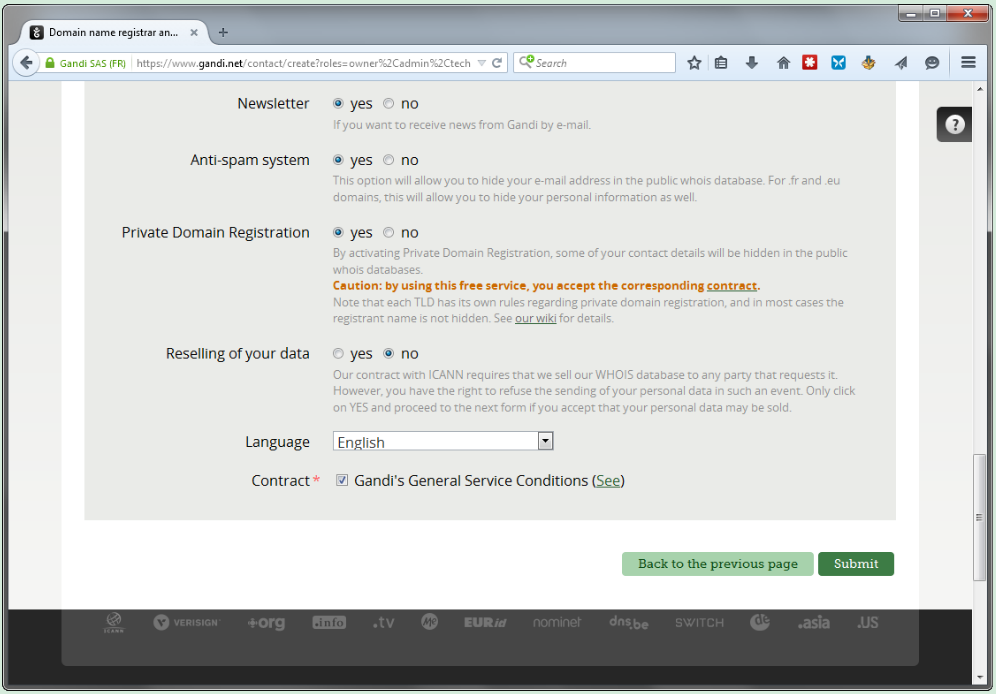
Ο domain registrar που προτείνει ο δημιουργός του Mail-in-a-Box είναι το Gandi.net. Ένα από τα θετικά του Gandi είναι ότι επιτρέπει τον ορισμό των λεγόμενων glue records ή αλλιώς child nameservers. Πρακτικά, με τα glue re-

cords δηλώνουμε ότι το VPS με το Mail-in-a-Box διαθέτει έναν πλήρη DNS server, ο οποίος αποτελεί μέλος του παγκόσμιου domain name system. Ένα άλλο καλό του Gandi είναι η υποστήριξη του DNSSEC – αλλά για συγκεκριμένα domains: http://wiki.gandi.net/en/domains/dnssec#who_can_use_dnssec. Περισσότερα για τα glue records και το DNSSEC θα δούμε στο τρίτο μέρος του αφιερώματός μας. Προς το παρόν, να σημειώσουμε ότι το Gandi υποστηρίζει μια μεγάλη πληθώρα TLDs (Top-Level Domains). Μετά από λίγο ψάξιμο καταλήξαμε στο –κάπως παράξενο, ομολογουμένως– colder.xyz (ναι, υπάρχει κατάληξη dot-xyz). Τις ημέρες που έγινε η αγορά, τα dot-xyz domains ήταν σε προσφορά: 2,50 ευρώ ανά έτος συν κάτι παραπάνω, λόγω φόρων και πολιτικών ελαχίστων χρεώσεων. Πήραμε λοιπόν το ολοκαίνουργιο domain μας και, όταν τελειώσαμε με την εγκατάσταση του Mail-in-a-Box, οι διευθύνσεις email του domain μας είχαν τη μορφή somebody@colder.xyz. Παρουσιάζουν μια κάποια πρωτοτυπία και τραβούν την προσοχή, δεν συμφωνείτε;

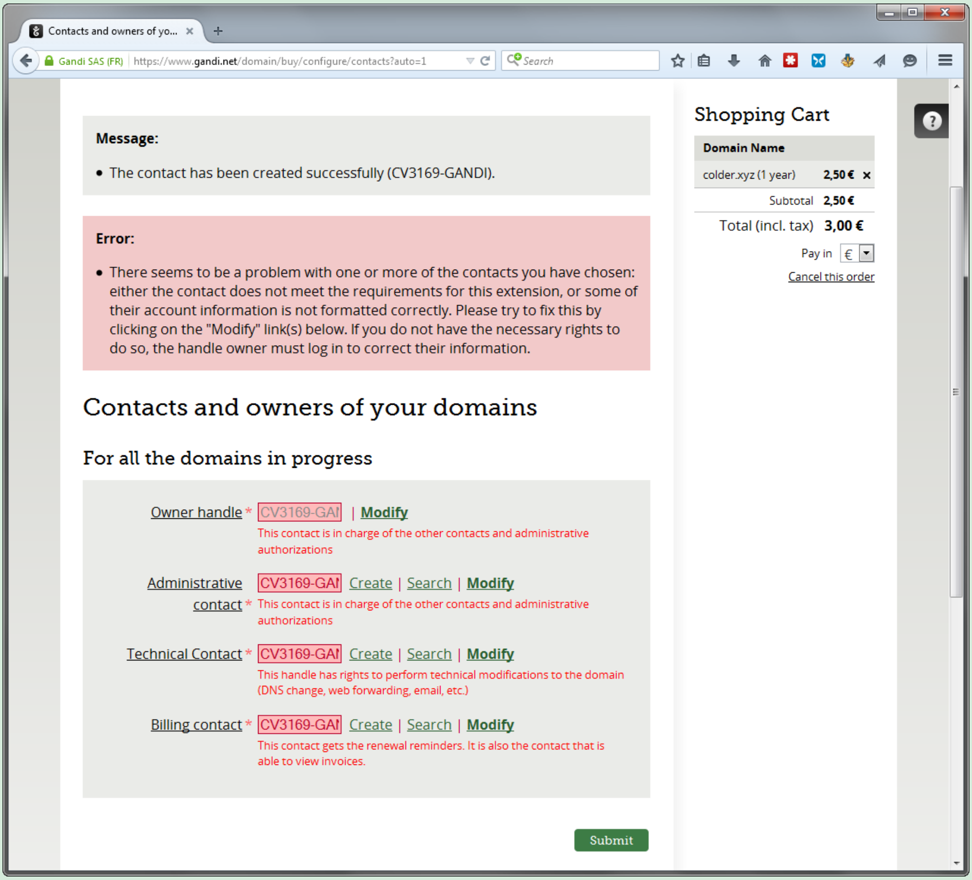
Δείτε στα ακόλουθα screenshots μερικά στιγμιότυπα από την αγορά του domain μας (και παρακαλούμε διαβάστε τις αντίστοιχες περιγραφές).



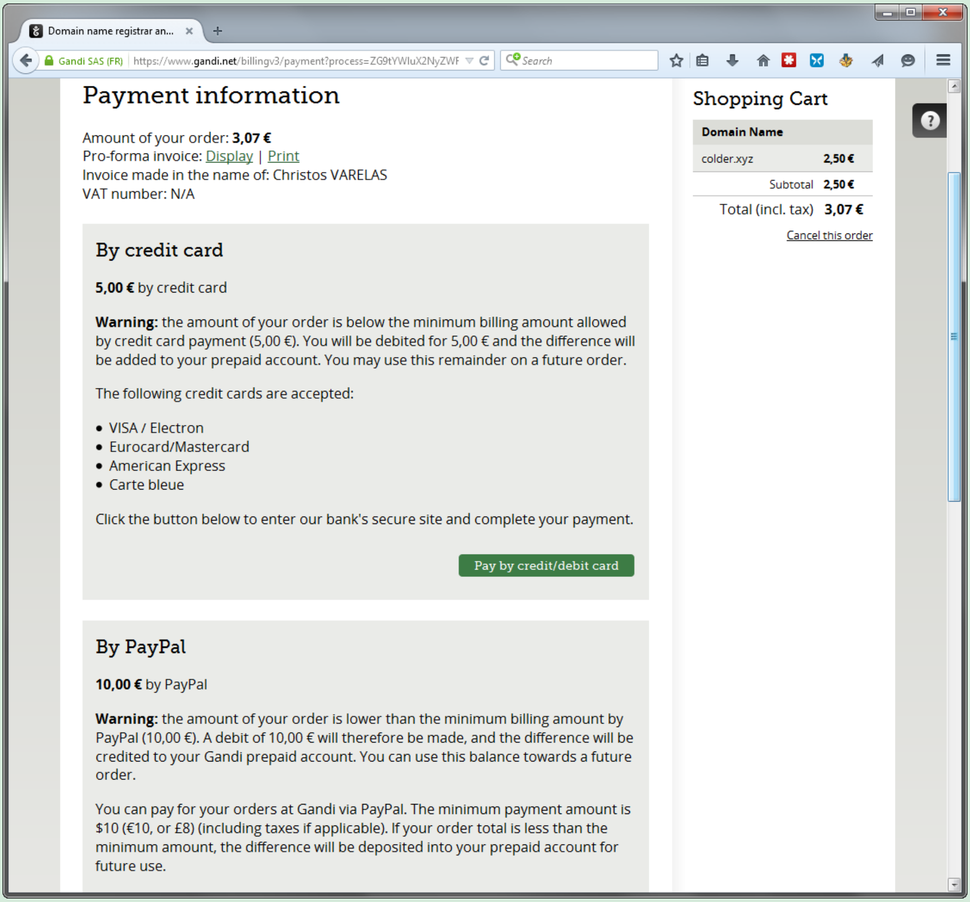
Μετά από λίγο ψάξιμο αποφασίσαμε ότι το colder.xyz είναι ό,τι πρέπει για το project μας. Όπως βλέπετε, μαζί με τους φόρους προσφερόταν στα 3 ευρώ ανά έτος. Η έκπτωση ίσως ισχύει και τη στιγμή που διαβάζετε το παρόν.



Με την αγορά νέου domain προτείνεται να ζητάμε την απόκρυψη από το WHOIS όσο το δυνατόν περισσοτέρων στοιχείων μας αφορούν. Διαφορετικοί registrars ονομάζουν διαφορετικά τη σχετική δυνατότητα. Στο Gandi επιλέξαμε τα Anti-spam system (απόκρυψη email) και Private Domain Registration (απόκρυψη στοιχείων επικοινωνίας). Αργότερα, πάντως, διαπιστώσαμε ότι το ονοματεπώνυμο εξακολουθεί να φαίνεται στο WHOIS. Παρατηρήστε εξάλλου ότι θέσαμε Reselling of your data = No, ώστε η εταιρεία να μη δικαιούται να πουλήσει τα δικά μας στοιχεία WHOIS σε όποιον τα ζητήσει. Ναι, ξέρουμε τι σκέφτεστε: χαμός γίνεται, Εκεί Έξω (TM).



Κατά τη συμπλήρωση των στοιχείων μας για την αγορά του ολοκαίνουργιου domain, το Gandi σκάλωσε και παραπονέθηκε: είτε η "επαφή" δεν πληρούσε τα προαπαιτούμενα για τη συγκεκριμένη κατάληξη, είτε οι πληροφορίες του λογαριασμού μας δεν εισήχθησαν σωστά. Μετά από λίγο ψάξιμο διαπιστώσαμε ότι είχαμε εισαγάγει τον ταχυδρομικό κώδικα (ΤΚ) σε μη-αποδεκτή μορφή. Συγκεκριμένα, το Gandi ήθελε τον ΤΚ με τη μορφή "abc xy", δηλαδή με ένα κενό μετά τα τρία ψηφία από αριστερά (και χωρίς τα εισαγωγικά).



Όταν επιχειρούμε να πληρώσουμε με πιστωτική, το Gandi δεν μπορεί να δεχτεί λιγότερα από 5 ευρώ. Όταν πάλι πληρώνουμε με PayPal, τότε θέλει τουλάχιστον 10 ευρώ. Σε κάθε περίπτωση, το υπόλοιπο πιστώνεται στο λογαριασμό μας και χρησιμοποιείται σε επόμενη αγορά.

Δημιουργία του κατάλληλου VPS

Για άλλη μια φορά επιλέξαμε τη DigitalOcean. Δεν είναι βέβαια υποχρεωτικό να φτιάξετε το VPS σας εκεί. Σημειώνουμε όμως πως αν το κάνετε, ακολουθώντας αυτό το referral URL → http://bit.ly/digocean10off, τότε κερδίζετε αυτομάτως 10$ σε credit. Επειδή το Mail-in-a-Box χρειάζεται VPS με τουλάχιστον 768MB μνήμης RAM, αναγκαστικά πηγαίνουμε στο δεύτερο κατά σειρά μεγέθους droplet, σ' αυτό δηλαδή που διαθέτει 1GB RAM. (Η DigitalOcean ονομάζει τα VPS της "droplets".) Το κόστος του ανέρχεται στα 10 δολάρια ανά μήνα, συνεπώς με το προαναφερθέν referral URL ο πρώτος μήνας είναι δωρεάν.

Η DigitalOcean προσφέρει δωρεάν DNS manager, αλλά δεν θα τον χρησιμοποιήσουμε. Θυμηθείτε ότι, εκτός από τον mailserver, το Mail-in-a-Box ρυθμίζει και DNS server, φροντίζοντας αυτόματα για όλα τα σωστά records ώστε αφενός ο mailserver του domain μας να λειτουργεί σωστά, αφετέρου να τον "σέβονται" οι άλλοι mailservers και να μένει μακριά από blacklists.

Μια άλλη δυνατότητα που παρέχει η DigitalOcean και μας είναι χρήσιμη, αφορά στον εύκολο ορισμό του λεγόμενου reverse DNS ή αλλιώς PTR record. Όταν ξεκινάμε από ένα domain name και προσπαθούμε να βρούμε την αντίστοιχη διεύθυνση IP, τότε κάνουμε το λεγόμενο forward DNS resolution. Το αντίστροφο, δηλαδή η αναζήτηση του domain name ξεκινώντας από μια διεύθυνση IP, ονομάζεται reverse DNS lookup ή αλλιώς reverse DNS resolution. Στο πρώτο βήμα της δημιουργίας νέου droplet στη DigitalOcean, καλούμαστε να πληκτρολογήσουμε ένα hostname. Το Mail-in-a-Box αναμένει αυτό το hostname να έχει τη μορφή

```
box.domain.tld
```

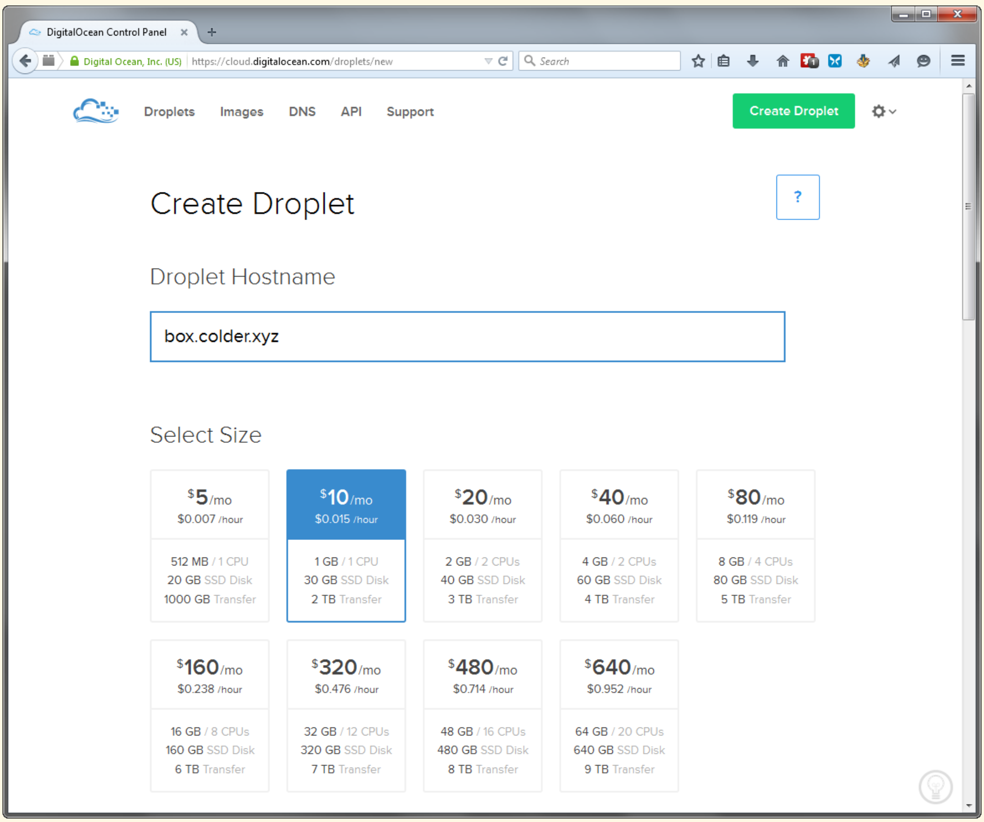
Παράδειγμα: Στη δική μας περίπτωση ισχύει domain.tld = colder.xyz, επομένως ως hostname για το droplet μας δώσαμε το

```
box.colder.xyz
```

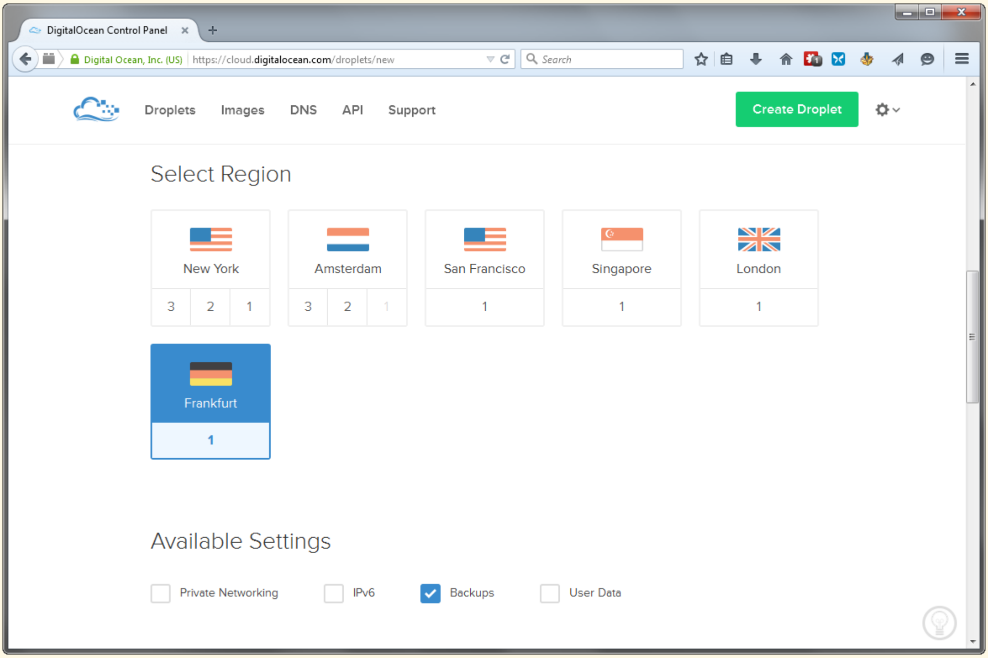
Αυτό ακριβώς το hostname χρησιμοποιείται αυτομάτως για τη δημιουργία του πολύτιμου PTR record. Έτσι, ένα reverse DNS lookup με το δημόσιο IP του droplet μας, ως αποτέλεσμα επιστρέφει το box.colder.xyz. Προσοχή: Επαναλαμβάνουμε ότι το Mail-in-a-Box περιμένει hostname της μορφής box.domain.tld, ενώ υποχρεωτικός είναι κι ο ανάλογος ορισμός του PTR record. Αν δεν χρησιμοποιείτε τη DigitalOcean, συμβουλευτείτε τις οδηγίες του αντίστοιχου cloud provider ώστε να θέσετε το PTR record όπως το αναμένει το Mail-in-a-Box.

Κάτι άλλο που οφείλουμε να σημειώσουμε είναι ότι, ανεξαρτήτως του cloud provider, το λειτουργικό σύστημα του VPS πρέπει να είναι το Ubuntu Server 14.04 LTS 64bit. Γι' αυτήν ακριβώς τη διανομή και έκδοση έχει γραφτεί το Mail-in-a-Box, οπότε δεν βλέπουμε το λόγο να διακινδυνεύσουμε την (αυξημένη) πιθανότητα για καζίκια,

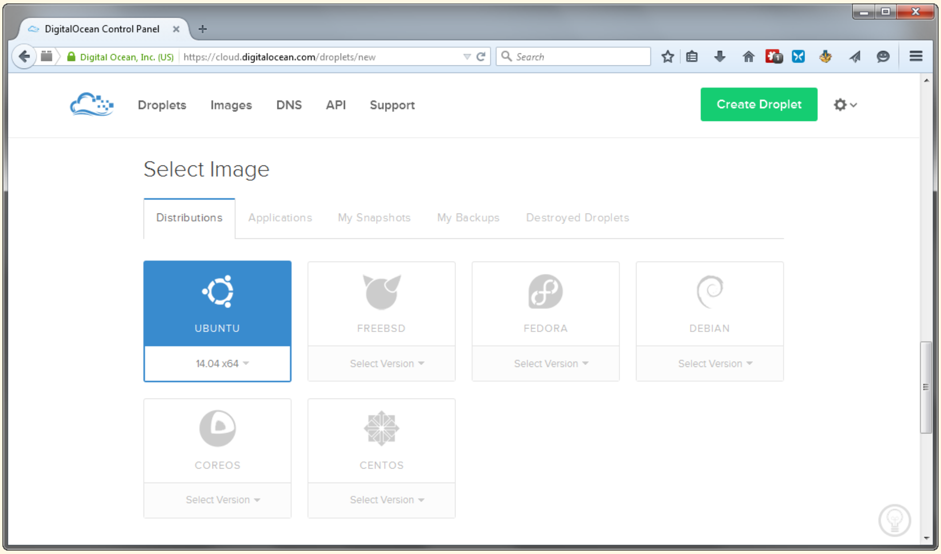
βάσανα και περιπέτειες, επιλέγοντας, π.χ., το Debian. Δείτε στα ακόλουθα screenshots χαρακτηριστικά στιγμιότυπα από τη δημιουργία του δικού μας VPS. Διαβάστε και τις αντίστοιχες περιγραφές – δεν είναι *τόσο* πολύ βαρετές.



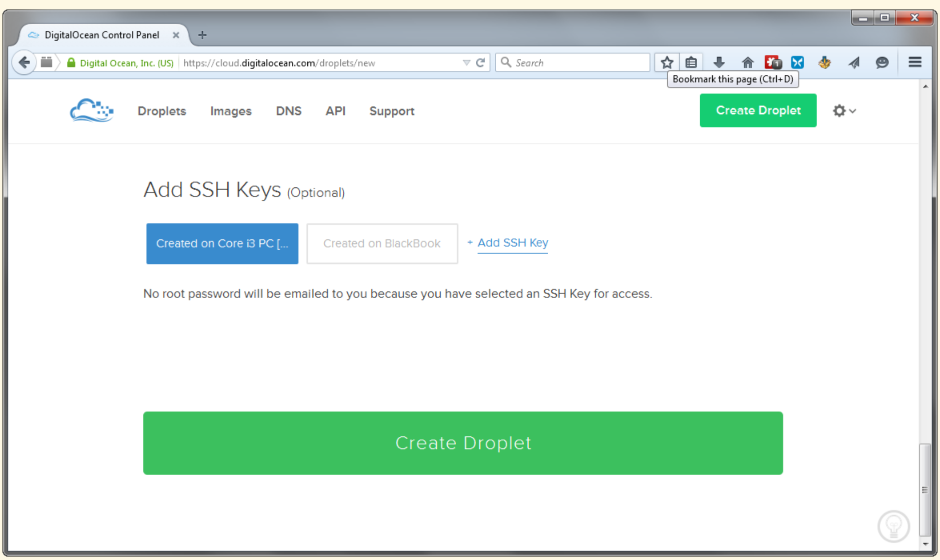
Το κατάλληλο droplet για να ξεκινήσουμε, αυτό με το οποίο το Mail-in-a-Box δεν θα έχει κανένα πρόβλημα, είναι ένα με μέγεθος μνήμης 1GB. Παρατηρήστε ότι καλούμαστε να πληκτρολογήσουμε ένα hostname. Το Mail-in-a-Box περιμένει να έχει τη μορφή box.domain.tld. Εμείς διαθέτουμε το domain ονόματι colder.xyz, οπότε το hostname του droplet μας αναγκαστικά γίνεται box.colder.xyz.



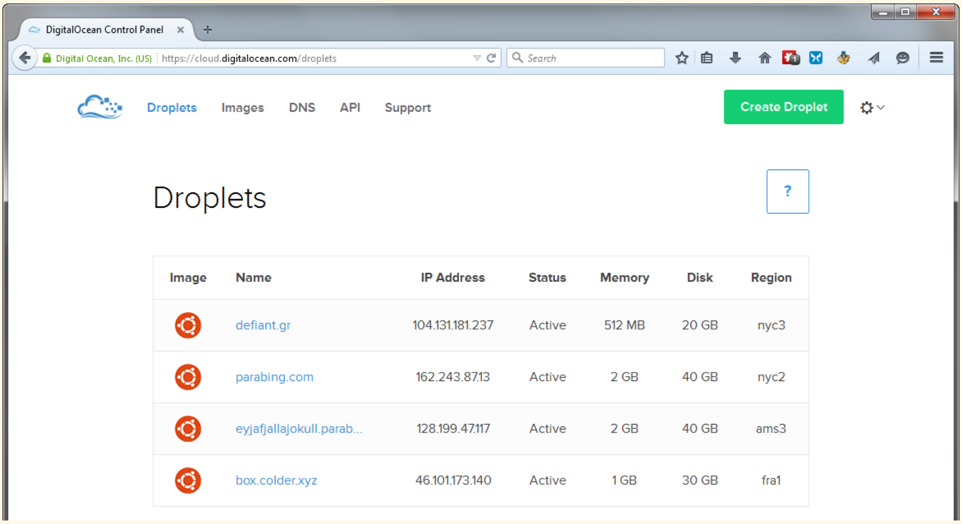
Εδώ επιλέγουμε περιοχή για τη δημιουργία του droplet. Είπαμε να πάμε με το datacenter της DigitalOcean στη Φρανκφούρτη, το οποίο αφενός είναι γεωγραφικά κοντά μας, αφετέρου είναι από εκείνα που υποστηρίζουν permanent resize operations. Γενικά, η ιδέα του resize είναι ότι μπορούμε να ξεκινάμε ένα νέο project με κάποιο μικρό και φθηνό VPS. Αν στο μέλλον αποφασίσουμε ότι χρειαζόμαστε ισχυρότερο (εικονικό) hardware, τότε αντί να δημιουργήσουμε καινούργιο VPS και να μεταφέρουμε σ' αυτό τις εφαρμογές μας (χρονοβόρα και συχνά δύσκολη εργασία), απλά "μεγαλώνουμε" το υπάρχον VPS και συνεχίζουμε μ' αυτό. Η DigitalOcean υποστηρίζει δύο ειδών resize operations: τα permanent και τα flexible resizes. Με permanent resize μεγαλώνουμε μνήμη, επεξεργαστή και δίσκο. Από την άλλη, με το flexible μεγαλώνουμε μόνο τη μνήμη και τον επεξεργαστή – όχι το δίσκο. Θέλαμε να έχουμε την επιλογή για permanent resize κι επειδή δεν υποστηρίζεται σε όλα τα datacenters της DigitalOcean, το ψάξαμε λίγο και διαπιστώσαμε ότι εκείνο στη Φρανκφούρτη τα υποστηρίζει. Θα παρατηρήσετε εξάλλου ότι έχουμε τσεκάρει την επιλογή για backups. Αυτά αφορούν σε ολόκληρο το droplet, είναι εβδομαδιαία και κάθε backup διατηρείται στους servers της DigitalOcean για τέσσερις εβδομάδες. Η ενεργοποίησή τους αυξάνει το κόστος του αντίστοιχου VPS κατά 20%, επομένως αυτό που φτιάξαμε από τα 10 δολάρια ανά μήνα ανεβαίνει στα 12 δολάρια ανά μήνα. Για να είμαστε ειλικρινείς, επιλέξαμε το backup της DigitalOcean από καθαρή περιέργεια – ίσως και για να πειραματιστούμε. Δεν σας προτείνουμε να το ενεργοποιήσετε κι εσείς. Αργότερα, άλλωστε, όταν θα έχουμε το Mail-in-a-Box έτοιμο και πλήρως λειτουργικό, θα διαπιστώσουμε ότι δημιουργεί κι εκείνο backups των email, τα οποία μάλιστα είναι incremental και κρυπτογραφημένα. Τα συγκεκριμένα αντίγραφα εφεδρείας δημιουργούνται τοπικά (στο droplet) και είναι δική μας ευθύνη να τα αντιγράφουμε κι αλλού. (Για να δικαιούμαστε να μιλάμε για σωστό backup πρέπει να έχουμε τουλάχιστον δύο αντίγραφα και το ένα εξ αυτών να είναι offsite.)



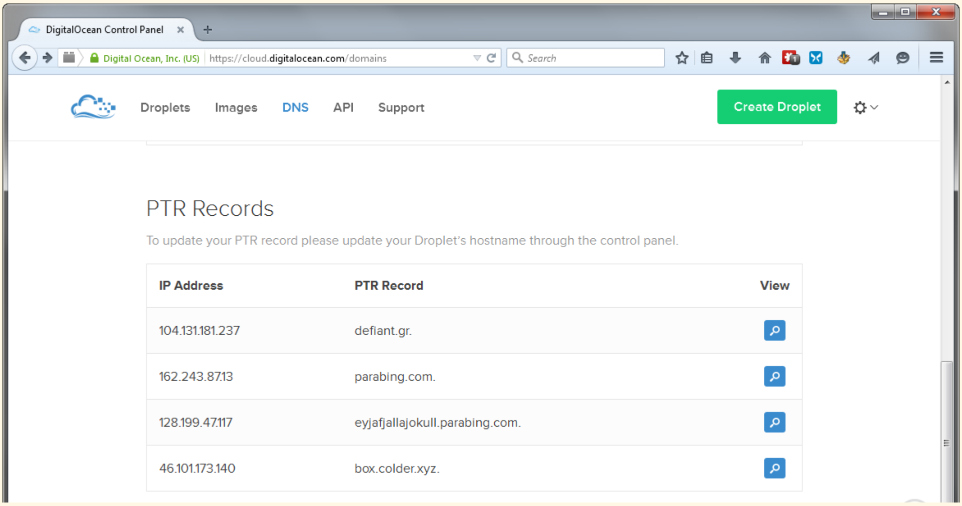
Το Mail-in-a-Box είναι γραμμένο για το Ubuntu Server και συγκεκριμένα για την έκδοση 14.04 LTS 64bit. Υποψιαζόμαστε ότι αυτό ίσως δυσαρεστήσει φίλους άλλων διανομών Linux. Αυτό. Δεν λέμε τίποτε άλλο.



Προσθέτουμε τα δημόσια κλειδιά των υπολογιστών που χρησιμοποιούμε, ώστε μετά την πρώτη εκκίνηση του droplet να είμαστε σε θέση για σύνδεση στο λογαριασμό του root μέσω SSH και χωρίς την πληκτρολόγηση password. Ορίζοντας SSH keys, δεν θα μας αποσταλεί καν κάποιο email με το password του root. Αν θέλετε να μάθετε περισσότερα για τα passwordless SSH logins, διαβάστε το σχετικό μας άρθρο στο http://deltahacker.gr/?p=13357.



Ιδού όλοι οι servers μας, στη DigitalOcean. Θα παρατηρήσετε ότι κάποιος λάμπει δια της απουσίας του! Πού θα πάει, κάποια στιγμή μέσα στο καλοκαιράκι θα τον μεταφέρουμε κι αυτόν, ενώ ο κος Αρχισυντάκτης (TM) θα περάσει και το νέο theme στο αντίστοιχο site :D



Όλα τα PTR records που αυτομάτως έχει θέσει η DigitalOcean για τα droplets μας. Όποιος, λοιπόν, κάνει ένα reverse DNS lookup για το 64.101.173.140, θα βρει ότι δείχνει στο box.colder.xyz. Τα antispam filters συνηθίζουν να πραγματοποιούν reverse DNS lookups και η ιδέα είναι ότι οι spammers δεν μπαίνουν στον κόπο να θέτουν PTR records. Σημειώστε ότι ο ορισμός των PTR records γίνεται από τον ISP (ή από τον VPS provider), ασχέτως αν έχουμε τον δικό μας DNS server για το domain ή όχι.

Βασικές ρυθμίσεις

Μετά την πρώτη εκκίνηση του VPS και την απομακρυσμένη σύνδεση μέσω SSH στο λογαριασμό του root, καλό είναι να ορίσουμε *αμέσως* ένα (νέο) συνθηματικό:

```
# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Πριν εκτελέσουμε οποιαδήποτε άλλη εργασία –και για να τελειώνουμε μ' αυτό–, ανοίγουμε το αρχείο /etc/hosts με τον text editor της προτίμησής μας και φροντίζουμε ώστε οι δηλώσεις για τα IPv4 hosts να είναι κάπως έτσι:

```
127.0.0.1       localhost.localdomain   localhost
46.101.173.140  box.colder.xyz          box
```

Η δεύτερη γραμμή θα διαφέρει στην περίπτωσή σας. Για την ακρίβεια, στη θέση του 64.101.173.140 θα πρέπει να βάλετε τη διεύθυνση IP του VPS σας: δείτε την από τον πίνακα ελέγχου της DigitalOcean ή απλά πληκτρολογώντας ifconfig – εννοείται σε ένα άλλο τερματικό από το οποίο έχετε συνδεθεί στο VPS. Επίσης, στη θέση του box.colder.xyz θα βάλετε το hostname του δικού σας VPS. Αποθηκεύστε τις αλλαγές στο /etc/hosts κι εγκαταλείψτε τον editor.

Εξ ορισμού, τα droplets της DigitalOcean δεν έχουν swap partition. Εύκολα όμως μπορούμε να φτιάξουμε ένα αρχείο αντιμετάθεσης – στο εξής swapfile. Το μέγεθος της μνήμης του VPS μας είναι 1GB κι ένας κανόνας λέει ότι το swapfile καλό είναι να 'χει το διπλάσιο μέγεθος. Ακολουθήσαμε τον κανόνα αλλά εις διπλούν, αφού κάτι μας έπιασε και σκεφτήκαμε ότι σύντομα θα κάνουμε resize το droplet. Τότε, θα έχει 2GB RAM (τόση μνήμη διαθέτει το αμέσως μεγαλύτερο droplet της DigitalOcean) και συνεπώς το swapfile που φτιάχνουμε τώρα ας έχει μέγεθος 4GB. Βέβαια θα μπορούσαμε να ξεκινήσουμε με ένα swapfile μεγέθους 2GB κι όταν αργότερα κάνουμε resize να το καταργήσουμε, να διαγράψουμε το σχετικό αρχείο και να φτιάξουμε ένα νέο swapfile, με μέγεθος 4GB. Ξέρετε κάτι, όμως; Εκεί που σκεφτόμασταν "swapfile το ένα", "swapfile το άλλο", ξαφνικά αναφωνήσαμε κάτι σαν "ινάφ γουίθ δε σουάπ-φάκινγκ-φάιλ ολρέντι" και χωρίς καμία άλλη καθυστέρηση φτιάξαμε ένα swapfile μεγέθους 4GB, για να ησυχάσουμε:

```
# fallocate -l 4G /swapfile
# chmod 600 /swapfile
# mkswap /swapfile
```

```
# swapon /swapfile
# echo "/swapfile none swap sw 0 0" >> /etc/fstab
```

Κάτι που δεν έχουμε κάνει ακόμα είναι να ενημερώσουμε την τοπική βάση με τα διαθέσιμα πακέτα για το Ubuntu, βεβαίως και ν' αναβαθμίσουμε εκείνα για τα οποία διατίθεται νεότερη έκδοση:

```
# apt-get update
# apt-get dist-upgrade
```

Πιθανώς να ενημερωθεί κι ο πυρήνας, οπότε θα χρειαστεί μια επανεκκίνηση. Μην την κάνετε ακόμα, να ορίσουμε πρώτα τη ζώνη ώρας και να εγκαταστήσουμε τον κλασικό NTP client, ώστε το ρολόι του συστήματος να ενημερώνεται αυτόματα από δικτυακούς time servers. Για τη ρύθμιση της ζώνης ώρας αρκεί να πληκτρολογήσουμε

```
# dpkg-reconfigure tzdata
```

ενώ για την εγκατάσταση του NTP client να δώσουμε

```
# apt-get install ntp
```

Μπορούμε τώρα να κάνουμε ένα reboot, έστω κι αν δεν χρειάζεται (σ' αυτό το αρχικό στάδιο, σίγουρα δεν βλάπτει):

```
# reboot
```

Πάμε τώρα να δούμε μερικές απολύτως απαραίτητες εργασίες που αφορούν στην ασφάλεια.

Ενίσχυση της ασφάλειας

Δεν είναι φρόνιμο να συνδεόμαστε απευθείας στο λογαριασμό του χρήστη root. Γενικά, δικαιώματα root πρέπει να αποκτούμε μόνον όταν είναι απαραίτητο και μετά καλό είναι να επιστρέφουμε στο λογαριασμό ενός απλού χρήστη. Στο Ubuntu αυτή τη στιγμή υπάρχει μόνον ο λογαριασμός του root. Ας συνδεθούμε λοιπόν σ' αυτόν για άλλη μία φορά, ώστε να δημιουργήσουμε το λογαριασμό ενός απλού χρήστη που θα μπορεί να εκτελεί το sudo και κατ' αυτόν τον τρόπο ν' αποκτά (προσωρινά) δικαιώματα διαχειριστή συστήματος. Στο παράδειγμα που ακολουθεί ο χρήστης μας έχει username το sub0. Ονομάστε τον δικό σας χρήστη όπως θέλετε.

```
# adduser sub0
Adding user `sub0' ...
Adding new group `sub0' (1000) ...
Adding new user `sub0' (1000) with group `sub0' ...
Creating home directory `/home/sub0' ...
Copying files from `/etc/skel' ...
Enter new UNIX password: εδώ_πληκτρολογούμε_συνθηματικό
Retype new UNIX password: εδώ_πληκτρολογούμε_συνθηματικό
passwd: password updated successfully
Changing the user information for sub0
Enter the new value, or press ENTER for the default
        Full Name []: subZraw
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
```

Επιτρέπουμε στο χρήστη sub0 να χρησιμοποιεί το sudo:

```
# gpasswd -a sub0 sudo
```

Πολύ ωραία. Δεν χρειάζεται να μένουμε άλλο στο λογαριασμό του root, οπότε όπως είμαστε ας μεταβούμε στο λογαριασμό του νέου μας χρήστη:

```
# su - sub0
```

Θέλουμε να συνδεόμαστε στο λογαριασμό του sub0 μέσω SSH και χωρίς να πληκτρολογούμε το password του:

```
$ mkdir ~/.ssh
$ chmod 700 ~/.ssh
$ touch ~/.ssh/authorized_keys
$ chmod 600 ~/.ssh/authorized_keys
```

Στο αρχείο authorized_keys του sub0 πρέπει να μεταφέρουμε τα δημόσια κλειδιά μας – κι ενδεχομένως μιλάμε για τα ίδια δημόσια κλειδιά που βρίσκονται στο authorized_keys του root. Αν ό,τι είπαμε δεν βγάζει νόημα, διαβάστε το επεξηγηματικό άρθρο στο http://deltahacker.gr/?p=13357.

Κάπου εδώ είναι σημαντικό να βεβαιωθούμε ότι έχουμε δυνατότητα για απομακρυσμένη σύνδεση στο λογαριασμό του sub0, μέσω SSH. Αν όλα είναι εντάξει, θα πρέπει να συνδεθούμε *χωρίς* την πληκτρολόγηση password. Και μετά από επιτυχή έλεγχο, μπορούμε να ζητήσουμε από τον OpenSSH server του Ubuntu να γίνει πιο αυστηρός ως προς τα είδη των συνδέσεων που επιτρέπει. Συγκεκριμένα, ας μη δέχεται καν συνδέσεις στο λογαριασμό του root, ενώ ας επιτρέπει *μόνο* τα passwordless logins, με χρήση κλειδιών SSH. Απλά ανοίγουμε το αρχείο /etc/ssh/sshd_config προς επεξεργασία και με δικαιώματα root

```
$ sudo nano /etc/ssh/sshd_config
```

και φροντίζουμε ώστε να περιλαμβάνει τις δύο ακόλουθες γραμμές:

```
PermitRootLogin no
PasswordAuthentication no
```

Αποθηκεύουμε τις αλλαγές, εγκαταλείπουμε τον editor κι επανεκκινούμε την υπηρεσία του OpenSSH:

```
$ sudo service ssh restart
```

Σε άλλες περιπτώσεις θα έχετε ίσως παρατηρήσει πως συνηθίζουμε να συνηγορούμε υπέρ της χρήσης ενός εναλλακτικού port για τον OpenSSH server (το προκαθορισμένο είναι το 22). Αυτή τη φορά δεν το κάνουμε: Σε λίγο που θα εγκαταστήσουμε το Mail-in-a-Box θα ρυθμιστεί αυτόματα το iptables ώστε να δέχεται συνδέσεις στο port 22/TCP, μ' άλλα λόγια συνδέσεις SSH. Επιπρόσθετα, θα οριστούν κανόνες που προστατεύουν τις υπηρεσίες του box από πιθανά DOS attacks, ενώ θα εγκατασταθεί και το fail2ban. Προς το παρόν, λοιπόν, δεν χρειάζεται να κάνουμε κάτι περισσότερο για την ασφάλεια του box. Το μόνο ίσως θέμα που αξίζει να δούμε είναι οι αυτόματες ενημερώσεις ασφαλείας. Μας αρέσει αυτή η δυνατότητα και σας προτείνουμε να την αξιοποιήσετε κι εσείς. Ανοίξτε με έναν text editor το αρχείο /etc/apt/apt.conf.d/10periodic και φροντίστε ώστε να περιλαμβάνει τις ακόλουθες γραμμές:

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";
APT::Periodic::Unattended-Upgrade "1";
```

Μετά ανοίξτε και το /etc/apt/apt.conf.d/50unattended-upgrades και βεβαιωθείτε ότι περιλαμβάνει μια οδηγία σαν αυτή:

```
Unattended-Upgrade::Mail "sub0@localhost"
```

Στο παράδειγμά μας, τα ενημερωτικά emails περί των αυτόματων ενημερώσεων παραδίδονται στον τοπικό λογαριασμό του χρήστη sub0. Όταν θα έχουμε έτοιμο τον mailserver μπορούμε να επανέλθουμε στο /etc/apt/apt.conf.d/50unattended-upgrades και να τροποποιήσουμε την οδηγία, ώστε τα emails να τα λαμβάνει κάποιος χρήστης του domain (π.χ., ο subZraw@colder.xyz).

Στο τρίτο μέρος του αφιερώματός μας, το οποίο δημοσιεύεται στο deltaHacker 044, εγκαθιστούμε και ρυθμίζουμε πλήρως το Mail-in-a-Box.


Ο ΔΑΣΚΑΛΟΣ ΑΝΕΒΑΣΕ ΜΙΑ ΚΑΙΝΟΥΡΙΑ ΣΕΛΦΙ ΣΤΟΝ ΤΟΙΧΟ ΤΟΥ!
ΝΤΑΞ... ΚΟΥΚΛΟΣ!
ΘΕΟΣ! <3 <3 <3
Sam fylakismenos eisai! lol
ΓΕΙΑ ΣΟΥ ΡΕ, ΞΑΔΕΡΦΕ!
ΣΩΡΑΙΟΣ!..
OMG!
ΠάNOS

deltahacker.gr